ESD-TR-70-373

MIXED-INITIATIVE SYSTEMS FOR TRAINING AND DECISION-AID APPLICATIONS

Jaime R. Carbonell
Allan M. Collins

November 1970

DEPUTY FOR COMMAND AND MANAGEMENT SYSTEMS
HQ ELECTRONIC SYSTEMS DIVISION (AFSC)
L. G. Hanscom Field, Bedford, Massachusetts 01730

(Prepared under Contract No. F19628-69-C-0298 by Bolt, Beranek and Newman, Inc., 50 Moulton Street, Cambridge, Massachusetts 02138.)

FOREWORD

One of the goals of Air Force Electronic Systems Division is the development of a technology for computer-based, personnel-support systems integrated into Air Force Information Systems. These support systems are required to improve the efficiency of man-computer interactions in the host Information Systems. They are designed to provide automated, on-the-job training and decision-aiding for Information System personnel.

Task 691703, Computer-Aided Instruction and Exercising Systems, under Project 6917, Command Management Data Systems Software, was established to develop and apply the technology for these personnel-support systems.

This report is one in a series supporting Project 6917. It describes a feasibility study for an entirely new approach to design of personnel-support systems. It presents an engineering solution based on artificial intelligence concepts and techniques. The unique data base structure results in significant improvements in individualization and humanization of the training and performance aiding process.

This work is a practical outgrowth of research on artificial intelligence techniques which has been funded for many years by the Office of Naval Research, Advanced Research Projects Agency, and Air Force Office of Scientific Research. The specific feasibility study and applications analysis presented here was supported by Contract F19626-69-C-0298.

Dr. Jaime R. Carbonell was the Principal Investigator. Dr. Sylvia R. Mayer, ESD/MCDS, served as Air Force Task Scientist and contract monitor. The effort was accomplished between June 1969 and August 1970.

This technical report has been reviewed and is approved.

Sylvia R. Mayer
SYLVIA R. MAYER
Task Scientist

William F. Heisler
WILLIAM F. HEISLER, Colonel, USAF
Director, Systems Design & Development
Deputy for Command & Management Systems

MIXED-INITIATIVE SYSTEMS FOR TRAINING AND DECISION-AID APPLICATIONS

Jaime R. Carbonell
Allan M. Collins

November 1, 1970

Electronics Systems Division
Laurence G. Hanscom Field
United States Air Force
Bedford, Massachusetts 01730

Attn: Dr. Sylvia R. Mayer

ABSTRACT

The project reported in this document presents SCHOLAR, the first prototype system capable of a true mixed-initiative man-computer dialogue on a given topic. The computer is not only capable of answering questions from the man (both using a comfortable and not too restricted subset of English), but also of generating questions, analyzing the man's responses, and producing reasonable consequent actions. All this occurs without full anticipation of conversational items and sequences. A very powerful program, applicable to many subject matters, acts upon a highly structured data base to generate the computer answers and questions, to evaluate the man's answers, and to produce suitable action sequences.

The U.S. Air Force, which has supported this research in part, can benefit considerably from the development of such mixed-initiative "knowledgeable" systems. The relevancy for applications like training, logistics and resource allocation, command and control systems, intelligence systems, and on-line design and planning is clear. A system built along the lines of SCHOLAR can be very valuable as an on-line aid to decision makers, by facilitating and guiding the interaction with complex and highly structured military data bases. SCHOLAR is also ideally suited to evolve into an on-line training facility to assist computer users faced with the need to efficiently utilize a new computer system or a new computer language.

The environment selected to develop SCHOLAR is in the field of training. SCHOLAR has been implemented as a new type of computer-assisted instruction (CAI) which results in what we can call information-structure-oriented (ISO) CAI systems. Some of the major features of ISO systems are: (1) they permit mixed-initiative dialogue; (2) the dialogue takes place in a comfortable subset of English; (3) the instructional programmer need not specify in advance all the questions, responses, and branchings that may occur; and (4) the mixed-initiative capabilities can be utilized to perform other tasks, such as that of an on-line decision aid.

This document covers both the theory supporting SCHOLAR and its implementation. Actual on-line protocols are used to illustrate the main features of the system.

TABLE OF CONTENTS

page

Abstract iii

Table of Contents iv

Table of Figures vi

I. INTRODUCTION AND OVERVIEW 1

I.1 The Goals and Results of the Project 1
I.2 A Brief Description of SCHOLAR 8
I.3 Modularity Transferability and Implementation Considerations 10

II. SCHOLAR AND OTHER CAI SYSTEMS 11

II.1 Traditional Approaches to CAI 11
II.2 ISO Versus AFO CAI 12
II.3 SCHOLAR's Capabilities 16

III. TECHNICAL DISCUSSION 18

III.1 Semantic Networks 18
III.2 Relevancy and Context 21
III.3 Natural-Language Man-Computer Communication .. 22
III.4 Questions: Their Nature and a Possible Taxonomy 23
III.5 Error Detection, Diagnosis, and Remedial Action 27
III.6 The Teacher Interactions in CAI 29
III.7 Overall Structure of SCHOLAR 33
III.8 Operation of SCHOLAR and Its Agenda 36

IV. CONCLUSIONS 39

IV.1 Some Implementation Considerations 39
IV.2 General Conclusions 41
IV.3 Recommendations for Further Work 43

APPENDIX A: THE CURRENT IMPLEMENTATION OF SCHOLAR 45

A.1 Overall System Organization 45
A.2 The Data Base 50
A.2a The Semantic Network 50
A.2b Other Permanent Information 55
A.2c Temporary Information 57
A.2d Some Auxiliary Procedures 61
A.3 Read, Print, and Other Interactive Procedures .. 62

page

A.4 The Retrieval Procedures 66
A.5 Processing Student Input 72
A.6 Generation of English Text 75
A.7 Overall Question-Answering Procedures 86
A.8 Question Generation and Context-Handling Procedures 88
A.9 The Matching Routines and Error Handling 97

APPENDIX B: GENERAL APPROACH IN DEVELOPING SCHOLAR 106

B.1 Some Pedagogical Considerations 106
B.2 Some Relevant Artificial Intelligence Questions 109

APPENDIX C: SOME MORE SCHOLAR PROTOCOLS 113

References 119

Bibliography 122

TABLE OF FIGURES

		page
Fig. 1 (a-e)	Fragment of an On-Line Protocol Using SCHOLAR	3
Fig. 2	Simplified Pictorial Representation of a Semantic Network on South America	9
Fig. 3	Trade Off Between AFO and ISO Systems in Terms of Data-Base Size	14
Fig. 4	The Units for Latitude and Argentina (Fragments) in SCHOLAR	19
Fig. 5 (a,b)	Simulated Protocol of a Teacher Building the Semantic Network	31
Fig. 6	General Structure of SCHOLAR	34
Fig. A.1	Schematic Flow-Chart of SCHOLAR	46
Fig. A.2	Principal Components in SCHOLAR	49
Fig. A.3	The Units for Latitude and Argentina (Fragments) in SCHOLAR	51
Fig. A.4	Approximate BNF Representation of the Syntax of SCHOLAR's Semantic Network	53
Fig. A.5	Semantic Irrelevancy for the Node ARGENTINA	56
Fig. A.6	The Context-Push Down List and the Used-Question List	60
Fig. A.7(a,b)	On-Line Protocol of Teacher Using SET INTERACTION in SCHOLAR	63
Fig. A.8	Triples in the Retrieval Package of SCHOLAR	67
Fig. A.9(a,b)	Structure of the Concept Unit "Height".....	69
Fig. A.10	Processing Student Input	73
Fig. A.11	English-Text Generator Operating on Answers About Uruguay	76

TABLE OF FIGURES

		page
Fig. A.12	Output of English-Text Generator and Internal Representation Related to a Complex Property	78
Fig. A.13 (a-e)	Traced Protocol of an Example of English-Text Generation	81
Fig. A.14	On-Line Protocol Using SCHOLAR's Question-Answering Mode	87
Fig. A.15	Handling of True-False Questions by SCHOLAR	89
Fig. A.16	Traced Protocol of an Example of Question-Answering by SCHOLAR	90
Fig. A.17	Traced Protocol of Context and Question Generation by SCHOLAR	94
Fig. A.18	Exhaustive Generation of Semantic Strings.	95
Fig. A.19	Question Generation in Different Modes from a Given Semantic String	96
Fig. A.20	Formation of WH Questions from Different Strings	98
Fig. A.21	Generation of Various True-False Questions by SCHOLAR	99
Fig. A.22	Generation of Various Multiple-Choice Questions by SCHOLAR	100
Fig. A.23	Matching Expected vs. Student Answers (Literal)	102
Fig. C.1	On-Line Protocol Using SCHOLAR's Question-Answering Mode	113
Fig. C.2 (a-c)	On-Line Mixed-Initiative Protocol	114
Fig. C.3 (a,b)	On-Line Mixed-Initiative Protocol	117

I.

INTRODUCTION AND OVERVIEW

I.1 The Goals and Results of the Project

The project reported in this document presents SCHOLAR, the first prototype system capable of a true mixed-initiative man-computer dialogue on a given topic. The computer is not only capable of answering questions from the man (both using a comfortable and not too restricted subset of English), but also of generating questions, analyzing the man's responses, and producing reasonable consequent actions. All this occurs without full anticipation of conversational items and sequences. A very powerful program, applicable to many subject matters, acts upon a highly structured data base to generate the computer answers and questions, to evaluate the man's answers, and to produce suitable action sequences.

The U.S. Air Force, which has supported this research in part, can benefit considerably from the development of such mixed-initiative "knowledgeable" systems. The relevancy for applications like training, logistics and resource allocation. command and control systems, intelligence systems, and on-line design and planning is clear. A system built along the lines of SCHOLAR can be very valuable as an on-line aid to decision makers, by facilitating and guiding the interaction with complex and highly structured military data bases. SCHOLAR is also ideally suited to evolve into an on-line training facility to assist computer users faced with the need to efficiently utilize a new computer system or a new computer language.

During the course of this development of a system capable of sustaining mixed-initiative dialogues with a man, a specific environment was needed; a convenient and efficient one seemed to be a verbally oriented training situation. In this restricted sense, this report introduces a new type of computer-assisted instruction (CAI), in many respects more powerful than existing ones, proves that it is feasible, and demonstrates by example some of its major capabilities. In the course of this investigation, a set of computer programs, the SCHOLAR system, was written(1). SCHOLAR is capable of reviewing the knowledge of a student in a given context by maintaining a mixed-initiative dialogue with him in a rather comfortable subset of English. The subject area selected for this application (geography of South America) is only a convenient one for demonstration purposes. It is convenient since its relational structure and item types are representative of a type of subject matter relevant to the Air Force, and in which there is a need for fast and low-cost techniques for developing expertise.

Examples of subject matters of the same type can be found within the areas of logistics, resource allocation, and intelligence.

Figure 1 (a to e) presents a fragment of a protocol, taken on-line, which demonstrates some of the basic capabilities of SCHOLAR. In this protocol, SCHOLAR starts typing after being called. The student's turn comes after the asterisk, and can be a response to a question by SCHOLAR, a question to SCHOLAR, or a command (like halting, or changing the mode of the interaction to either Q/A, for question-answering, or mixinit, for mixed-initiative, or test, for testing). The student returns control to SCHOLAR by typing another asterisk and a carriage return.

Observe that SCHOLAR can prompt the student, indicate when it does not understand him, detect misspellings, and answer the student's questions using acceptable English. SCHOLAR can also generate questions, and evaluate the student's answers, deciding when these are correct, wrong, or only approximately or partially correct, and then take some conditional actions. It keeps track of content, and changes it on the basis of relevancy and time considerations. SCHOLAR does all this without faithfully following specific and detailed directions, but rather by applying general criteria and procedures to a body of "knowledge" (see below) about the subject being discussed.

Traditional tutorial CAI systems depend for their operation on the utilization of blocks of material, usually called "frames", textually entered in advance by the teacher. We call such systems ad-hoc-frame-oriented (AFO) CAI systems. AFO systems require the teacher to prepare in advance the questions, answers, keywords, and branchings of the system. The student in AFO systems can not ask questions or use natural language in talking to the system. These limitations of AFO systems are a major justification for development of information-structure-oriented (ISO) systems. ISO systems, such as SCHOLAR, are designed so that the teacher puts in the basic knowledge he wants to transmit, and the program generates the responses, questions, branchings, and error corrections from its stored knowledge. For the student this permits him to ask questions when he doesn't understand something. He can converse with the system as with a tutor, a mode of teaching that is the best we know today. Another major reason for the development of ISO systems is that they can also be utilized for applications where a user needs an on-line decision aid. These considerations are discussed more fully in Section II.

To summarize the major features of ISO systems: (1) they permit mixed-initiative dialogue; (2) the dialogue takes place in a comfortable subset of English; (3) the teacher need not specify in advance all the questions, responses, and branchings that may occur; and(4) the mixed-initative capabilities can be utilized to perform other tasks, such as that of an on-line decision aid.

```
←SCHOLAR)

++++++++++++++++++++++++++++++++++++++++++++++++++++++++


 HELLO, MY NAME IS SCHOLAR. I WILL HELP YOU IN REVIEWING
THE GEOGRAPHY OF SOUTH AMERICA. THIS IS ABC REGIONAL
HIGH SCHOOL. TODAY IS 04/29/70.



 IF YOU HAVE CAREFULLY STUDIED YOUR INSTRUCTIONS, WE
MAY BEGIN THE REVIEW NOW.


 WHEN YOU ARE READY TO START, TYPE READY --- REMEMBER
TO TERMINATE WITH AN ASTERISK * FOLLOWED BY A CARRIAGE
RETURN.


*READY*

 PLEASE TYPE YOUR NAME --- MR., MISS, OR MRS. FOLLOWED
BY FIRST NAME, MIDDLE NAME OR MIDDLE INITIAL, AND LAST
NAME.

*MR. JAMES R. JONEES↑S↑ES*

 NAME OF YOUR INSTRUCTOR, PLEASE:

*MR. BROWN*


 LET'S BEGIN OUR DISCUSSION NOW, MR. JONES.


++++++++++++++++++++++++++++++++++++++++++++++++++++++++


 WHAT IS CHILE?

*A COUNTRY*

 THAT'S FINE.


 SELECT AN ALTERNATIVE FROM THE LIST:

 214000.
 800300.
 49883.
 63690.

 FOR THE QUESTION:

   APPROX WHAT IS THE AREA OF CHILE?
```

Fig. 1 (a) Fragment of an On-Line Protocol Using SCHOLAR (CONT.)

```
*800300*

 YOUR ANSWER IS INCORRECT.
 THE CORRECT ANSWER IS: 214000.


 THE POPULATION IN CHILE IS APPROX 8500000. PEOPLE.
 TRUE OR FALSE?

*TRUE*

 YOU ARE CORRECT.


 USE ONE OF THE FOLLOWING:

 SUCRE
 FRENCH
 ARGENTINA
 SPANISH

 TO ANSWER THE QUESTION:

   WHAT IS THE LANGUAGE IN CHILE?

*
 YOU ARE TAKING TOO MUCH TIME...

*SPANICH*

 YOU SHOULD HAVE WRITTEN "SPANISH".

 VERY GOOD.


 THE CAPITAL IN CHILE IS SANTIAGO.
 CORRECT OR INCORRECT?

*CORRECT*

 THAT'S RIGHT.

   APPROX WHAT IS THE AREA OF BRAZIL?

*2500000 SQUARE MILES*

 WRONG.
 PLEASE INDICATE IF THE FOLLOWING STATEMENT IS CORRECT
           OR INCORRECT:

 THE AREA OF PARAGUAY IS APPROX 47432 SQUARE MILES.

*WHAT IS THE AREA OF BRAZIL?*


   THE AREA OF BRAZIL IS APPROX 3600000. SQUARE MILES.

 NOW ANSWER THE QUESTION YOU DIDN'T ANSWER BEFORE.

*FALSE*
```

Fig. 1 (b) Fragment of an On-Line Protocol Using SCHOLAR (cont.)

```
REPHRASE YOUR STATEMENT. I CAN'T UNDERSTAND IT.

*INCORRECT*

VERY GOOD.

THE AREA OF PARAGUAY IS APPROX 157000 SQUARE MILES.


THE LANGUAGE IN BRAZIL IS FRENCH.
RIGHT OR WRONG?

*WRONG*

YOU ARE CORRECT.

THE LANGUAGE IN BRAZIL IS PORTUGUESE.
IS IT TRUE THAT THE CAPITAL IN BRAZIL IS BRAZILIA?

*YES*

THAT'S FINE.

  APPROX WHAT IS THE POPULATION IN BRAZIL?

*60000000.*

APPROXIMATELY TRUE, BUT A MORE EXACT VALUE IS 70000000.

WHAT IS BRAZIL?

*TELL ME SOMETHING ABOUT PERU.*


WAIT A MINUTE. I'M CHECKING MY FILES.

LET'S PROCEED NOW.


     PERU IS A COUNTRY.
     IT IS IN SOUTH AMERICA.
     THE POPULATION IS APPROX 7000000. PEOPLE.
     THE CAPITAL IS LIMA.

NOW ANSWER THE QUESTION YOU DIDN'T ANSWER BEFORE.

*Q/A*


DO YOU WANT TO CHANGE THE GENERAL MODE OF THE INTERACTION
     TO Q/A?
PLEASE ANSWER Y OR N.

*Y*

OK, MR. JONES. LET'S GO AHEAD.


*PLEASE, TELL ME MORE ABOUT PERU.*

     THE AREA OF PERU IS APPROX 480000. SQUARE MILES.
     THE LANGUAGE IS SPANISH.
```

Fig. 1 (c) Fragment of an On-Line Protocol Using SCHOLAR (cont.)

```
*HOW MANY SQUARE MILES ARE THERE IN BRAZIL?*

   THE AREA IS APPROX 3600000. SQUARE MILES.

*WHAT IS THE PRINCIPAL RELIGION IN URUGUAY?*

   THE PRINCIPAL RELIGION IN URUGUAY IS CATHOLICISM.

*
 WELL, HAVE YOU MADE UP YOUR MIND?

*THE PRINCIPAL RELIGION IN PERU IS ---.*

 I'M AFRAID I CAN'T ANSWER THAT.

*IS IT TRUE THAT THE AREA OF URUGUAY IS 70000?*
 THAT'S ABOUT RIGHT. THE EXACT ANSWER IS 79000.


*COULD YOU PLEASE INDICATE THE PRINCIPAL RIVERS IN
 ARGENTINA?*

 I DON'T UNDERSTAND YOUR WORD "INDICATE". I'LL GIVE
       YOU ANOTHER CHANCE.

*THE PRINCIPAL RIVERS IN ARGENTINA ARE ---.*

   THE PRINCIPAL RIVERS IN ARGENTINA ARE THE RIO DE LA PLATA,
       THE PARANA, THE URUGUAY RIVER, THE SALADO, THE
       PARAGUAY RIVER, AND THE PILCOMAYO.

*MIXINIT*

 DO YOU WANT TO CHANGE THE GENERAL MODE OF THE INTERACTION
      TO MIXINIT?
 PLEASE ANSWER Y OR N.

*Y**

 OK, MR. JONES. LET'S GO AHEAD.

 WHERE IS BRAZIL?


 PLEASE HURRY UP. I'M WAITING FOR YOUR STATEMENT.

*IN SOUTHAMERICA*

 I BELIEVE YOU MEANT TO TYPE "SOUTH AMERICA".

 THAT'S RIGHT.
```

Fig. 1 (d) Fragment of an On-Line Protocol Using SCHOLAR (cont.)

```
IS IT CORRECT TO SAY THAT THE CITIES IN BRAZIL ARE
    SAO PAULO, RIO DE JANEIRO, BRAZILIA, PORTO ALEGRE,
    BAHIA, RECIFE, SANTOS, MANAOS, AND BELEM?

*YES*

THAT'S FINE.

THE CITIES IN BOLIVIA ARE ----.

*LA PAZ, SUCRE, BOGOTA, AND ACONCAGUA*

AT LAST! ANYWAY, LET US SEE WHAT YOU HAVE JUST WRITTEN.

HAVEN'T YOU INCORRECTLY USED "BOGOTA" AND "ACONCAGUA"?
ON THE OTHER HAND "LA PAZ" AND "SUCRE" IS CORRECT.

THE COMPLETE ANSWER ALSO INCLUDES POTOSI, COCHABAMBA,
                              AND SANTA CRUZ.


SELECT AN ALTERNATIVE FROM THE LIST:

845600.
1033500.
-8297216.
4000000.

TO FILL THE BLANK BELOW:

THE POPULATION IN BOLIVIA IS APPROX --- PEOPLE.

*40
00 YOU ARE TAKING TOO MUCH TIME...

*4000000.*

THAT'S RIGHT.

TYPE EITHER T OR F AFTER THE FOLLOWING STATEMENT:

THE LANGUAGE IN BOLIVIA IS SPANISH.

*WHAT IS CORDOBA?*


     CORDOBA IS A CITY.
     IT IS LOCATED IN ARGENTINA.
     THE POPULATION IS 700000. PEOPLE.

NOW ANSWER THE QUESTION YOU DIDN'T ANSWER BEFORE.

++++++++++++++++++++++++++++++++++++++++++++++++++++++++

TODAY'S REVIEW IS OVER.
IT HAS BEEN A PLEASURE TO WORK WITH YOU, MR. JONES.

++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Fig. 1 (e) Fragment of an On-Line Protocol Using SCHOLAR (CONCLUDED)

I.2 A Brief Description of SCHOLAR

SCHOLAR has a data base that is a complex but well-defined information structure in the form of a network of facts, concepts and procedures. The elements of this network are units of information defining words and events in the form of multi-level tree-lists. The elements of those lists are other words which in turn point to their respective units, and so on. Figure 2 is a simplified pictorial representation of a portion of a network of this sort in the context of geography of South America. Each rectangle or plane is a unit with a name (Uruguay, Argentina, South America, country, latitude) and a set of symbolically coded properties. This kind of network is called a "semantic" network. Semantic networks were first introduced by the pioneering work of Quillian (2 and 3). No specific pieces of text, or questions with their predicted answers, errors, and anticipated branching form part of SCHOLAR's information structure, as is the case in the data bases of AFO CAI systems.

Instead SCHOLAR utilizes an executive program which is capable of probing the semantic network in order to generate out of it the material to be presented, and the questions to be asked to the student. This program is almost completely independent of the subject matter to which it applies. As shown in Fig. 1, this program is at the same time capable both of generating the corresponding answers to its own questions, and of a certain degree of branching conditional on the student's responses, while maintaining a continuity of contexts and subcontexts.

Furthermore, the data base of ISO CAI systems reflects basic "knowledge" about the subject under discussion; therefore (as shown again in Fig. 1), SCHOLAR can at any time accept questions from the student, thus using its semantic network for question-answering purposes.* This explains why mixed-initiative dialogues between man and computer are now possible, and why SCHOLAR can be used as a decision aid in on-line performance systems. The use of a semantic network also facilitates the two-way communication in a rather large and free subset of English.

SCHOLAR incorporates some components which, as far as we know, have not been developed before in computer systems, namely, the handling of relevancy and context, and the contextual generation of questions and text. These new features enable the system to produce true mixed-initiative dialogue for the first time. SCHOLAR is also something rather unusual in being a balanced integrated system, which is operational, and which incorporates as component subsystems different packages usually developed in isolation. A more detailed discussion of SCHOLAR from an artificial-intelligence point of view is presented in Appendix B.

---

*In other words, a question-answering system is a component of SCHOLAR.

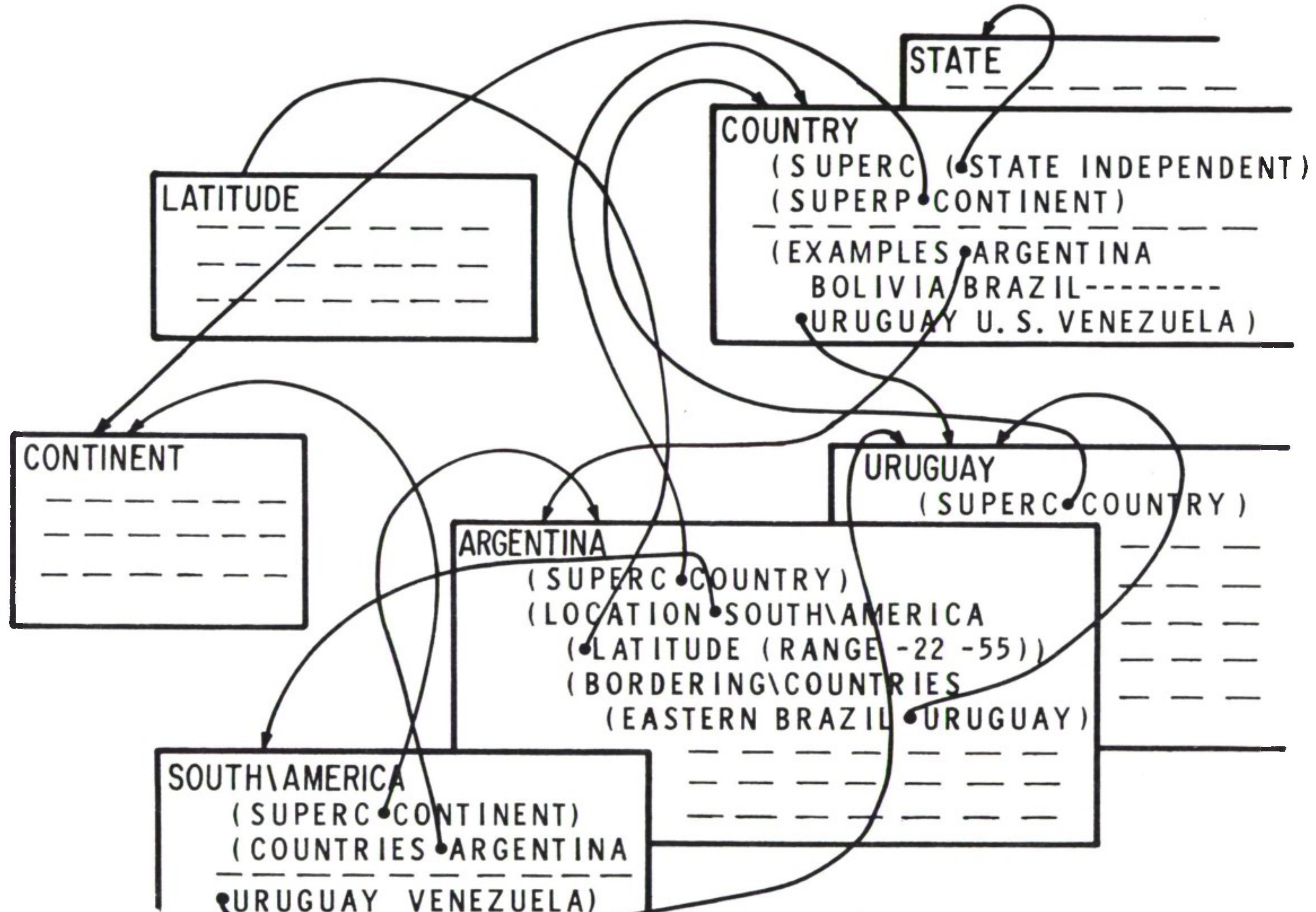

Fig. 2 Simplified Pictorial Representation of a Semantic Network on South America

I.3 Modularity Transferability and Implementation Considerations

SCHOLAR is the product of a systems-oriented effort in which we have balanced the development of the different components to achieve a demonstration of our approach within available resources in terms both of computer capabilities and development time. The modular construction of SCHOLAR permits extensions and even complete revisions of some portions with only minor effects on others.

The subject matter, geography of a given region, was selected as being representative of verbally oriented subjects with comparatively little inherent logical structure and contextual algorithms. Changing the example to which geography is applied presents no problem to SCHOLAR since only part of the semantic network must be updated. Changing the application to, say, anatomy, would mean an almost complete revision of the content of the semantic network, but not of the program. Shifting to a more computational or algorithmic topic (like aspects of Spanish syntax, or analytic geometry) would still require practically no changes in the program but would imply a semantic network much richer in procedures than that of more descriptive subjects (procedures like, for example, "conjugate" for Spanish verbs, or "intersection" for analytic geometry).

SCHOLAR is currently implemented in BBN-LISP in an XDS-940 time-sharing computer. This environment was principally selected because of its rather large virtual memory, obtained through paging. Conversion to a larger and faster Digital Equipment Corporation PDP-10 time-sharing system suitably modified to support paging is under way. SCHOLAR could be inplemented in computer systems with large real or virtual fast-access memory. Fast-access memories now being developed will be available at low cost in the near future.

In the remainder of this report we will present a discussion of how SCHOLAR fits into the general context of CAI systems, a top-level description of how SCHOLAR works, and finally our evaluation of what are the possible uses of SCHOLAR now and in the future. The first appendix presents the details of SCHOLAR's implementation; the second discusses SCHOLAR in relation to educational questions and artificial intelligence work; and the third shows additional protocols produced by SCHOLAR.

II.

SCHOLAR AND OTHER CAI SYSTEMS

The research work reported here and leading to the development of SCHOLAR is not strictly a part of any previously existing field of research. It is a rather new effort of an interdisciplinary nature. In the following section we will discuss SCHOLAR's relation to classic AFO CAI, and in particular to the Computer-Directed Training System (CODIT) (4) developed for the Air Force.

II.1 Traditional Approaches to CAI

Computer-assisted instruction efforts have in the past few years proceeded along several lines. Frye (5) and Zinn(6) have described existing systems and languages for CAI and attempted a taxonomy of these efforts. Bryan's(7) similar classification distinguishes three broad categories. In the first, ad-lib CAI, the student is given access to a computer (including one or more languages and perhaps a library of routines), but he is in full control; his input is not controlled by the computer. LOGO, developed by Feurzeig and Papert (8) is one of the interesting efforts of the ad-lib kind.

The second category is games and simulation, where the student has some initiative but is constrained by the rules of the game or the logic of the simulation. The Socratic Systems (Swets and Feurzeig,(9) is a program where all possible branches in a huge tree of alternatives (with possible loops) must be specifically programmed. That tree refers to an example of some diagnostic process (medical or otherwise) which the student must perform.

The third category in Bryan's taxonomy is called controlled learning and implies detailed anticipation and branching in a Crowderian sense. In general, these programs involve the construction of frames entered as text in advance by the teacher. The material in the frames usually includes paragraphs of English text to be presented, specific questions with their correct answers, some expected incorrect answers or keywords, and the anticipated branching for this limited set of alternative student answers. Questions are usually multiple-choice questions. We have called these systems "ad-hoc-frame-oriented" (AFO) CAI systems. PLANIT, discussed by Feingold(10), and ELIZA, described by Taylor(11) are among the well-known systems of this kind. The Air Force's CODIT is another system of this type.

In that SCHOLAR's goal is tutorial, it is most closely comparable to AFO CAI systems such as CODIT.

A system such as CODIT has many specific advantages to recommend it as a tutorial system:

(a) The student can proceed at his own rate.

(b) Working with a computer is highly motivating and gives the student familiarity with using the computer.

(c) The computer can compute overall statistics on the student's progress such as percent correct and average response time to questions.

(d) The student is an active participant in learning rather than a passive listener.

(e) The student gets immediate feedback as to how he is doing, and the errors he generates are corrected directly.

(f) The material taught to students is well specified so that it can easily be improved or refined depending on its success in training students.

(g) Materials prepared for programmed instruction can easily be converted to an AFO system.

However, there are several basic limitations to all AFO CAI systems. The student has little or no initiative; he cannot use natural language in his responses, and systems usually look fairly rigid to him. The teacher has a considerable burden in the preparation of questions, answers, keywords, and branching. From a systems point of view, the system controls the student but is in turn tightly ad-hoc programmed by the teacher; the system has no real initiative or decision power of its own; and, of course, it has no real "knowledge".

II.2 ISO Versus AFO CAI

We are now in a good position to establish a comparison between the classic ad-hoc frame-oriented CAI systems, and the information-structure-oriented ones we have just introduced. SCHOLAR is the first prototype of the latter kind, while CODIT is an example of the AFO kind.

Let us first consider the capabilities of both types of systems. Both can present material and questions to the student. AFO systems can at this time ask more involved questions but these must have been formulated in all detail in advance by a human teacher. ISO systems have better capabilities for analyzing unanticipated answers, which they can related to their semantic memory. Because of this, ISO systems can be designed to have diagnostic capabilities which AFO systems can not possess; they can only work on specific errors anticipated by the teacher.

AFO systems do not allow students' questions; ISO systems can handily process and answer them. This leads to a true conversational capability, with questions from both sides that will depend on overall context, specific context, what has just happened in the previous question, etc.

Because ISO systems have capability to answer questions, their knowledge about any particular subject matter can be put to other uses. With little modification an ISO system could be used as a question-answering system for fact retrieval, or computer aided design or command - and - control operations. One particular use we plan to implement in the future is an on-line decision-aiding system for computer users who have questions about how the system works. Other kinds of decision-aiding applications are also possible, but no such extensions are feasible with AFO systems.

The teacher preparing frames of text, questions, answers, and branching for AFO systems is faced with an extensive and rather unchallenging task. It is known that teachers preparing AFO CAI courses can barely catch up with the students which use up the material very fast. Preparing the 1000th question takes the same time and effort as preparing the first. Finally, in AFO systems the instructional programmer is not necessarily led to conceptualize his subject, since an AFO program does not call for an explicit structuring of the content material. On the contrary, the teacher's role in an ISO system is a more conceptual one, with less concern for repetitive examples. Adding a new piece of information to the data base usually permits many possible new questions; the program can also use that piece of information to draw inferences and set relations. The larger the semantic network on a given context the greater will be the effect of an addition to that network.

On the other hand, ISO systems require, for their proper operation, a larger data base than AFO systems. This means that, initially, the effort necessary to support, say, one-hour of console interaction is larger in ISO systems than in AFO systems. Putting information in the data base initially may also require more specialized personnel. This is so because of the need to enter general concepts (and their properties) about the field being discussed (like latitude, or temperature, in the case of geography). But later on, these concepts will be utilized many times without requiring further definition; this is where the ISO approach will become efficient. Figure 3 graphically depicts these trade-offs.

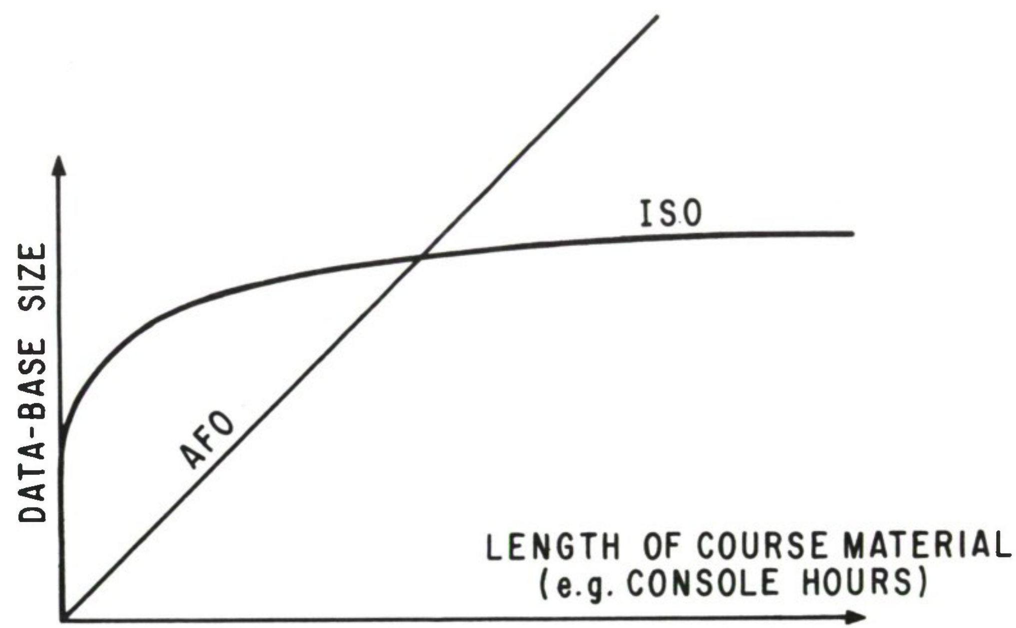

FIG.3 TRADE OFF BETWEEN AFO AND ISO SYSTEMS IN TERMS OF DATA-BASE SIZE

There is another aspect of putting information into the data base that deserves some discussion. There are restrictions as to what information must be put into the ISO system (such as the part of speech for any word entered) and also some conventions as to the format for entering information. To a certain extent ISO systems can be programmed to help the teacher enter information in the right format by dialogue with the teacher. Nevertheless, there probably will be more conventions for the teacher to learn in entering information into ISO systems than there are presently with AFO systems. On the other hand, these conventions are probably not harder to learn than the usual AFO conventions with respect to anticipated answers, keywords, and branching.

The sizable data base that ISO CAI systems require, for their minimal operation must also be readily available at all times. This data base must be in core or in a random-access device (like a drum) where it is accessible through paging. A semantic network (on which ISO systems are based) is not amenable to serial secondary storage devices like tape drives. These may be used, however, by AFO systems based on serial presentation of largely unrelated frames of information. Computer systems with small core and only serial secondary storage will not allow ISO CAI which is geared towards the large system. AFO CAI demands less from the computer because it offers less to the user. It makes a more modest, less sophisticated use of the power of a computer, and, the reader will agree, provides a more modest instructional capability. ISO CAI is much more in tune with the current state of the art in computer science, and because of this, is capable of producing considerably more interesting instructional interactions.

As to the question of costs, we believe that a cost comparison between AFO and ISO CAI is premature at this stage. The ISO approach is only in the initial stage of development, and SCHOLAR is the only pilot system in existence. In general terms, though, it seems clear that AFO systems, offering less, will cost less than related ISO systems. But it is possible to argue that, when teachers time and effort is included, and when educational objectives are taken as unitary measures, ISO systems might be, for certain applications and in a not too distant future, quite competitive.

In terms of practical realization and use with students, ISO systems will still be objects of continued development for some time to come. AFO systems can be and are being implemented in many computers now. Actually, the problem with them is that they are frequently using facilities too powerful for what those systems require. ISO CAI systems, on the other hand, make heavy (and balanced, we believe) use of the different computer components, i.e., memory, central processing, and input-output devices. In order for ISO systems to be practical, they will require, perhaps, more powerful computational facilities than those existing now, especially in terms of fast-access memory. In fact, low-cost fast-access memories are currently being developed.

Let us conclude these brief remarks by emphasizing that we are not advocating the replacement of AFO by ISO systems. AFO systems will have their role for some time to come. We see them convenient for cases in which the subject matter is very diversified or unstructured and the interactions with the students are planned to be brief. In that case the development of complex semantic networks is not justified. When discussion in depth is desired, or both the student and the computer should have initiative, or when detailed anticipation is unwanted, then ISO systems are to be preferred. Beyond this, extensions to decision-aiding applications is only possible with ISO systems.

II.3 SCHOLAR's Capabilities

At this point it seems appropriate to list what we feel are the major advantages of SCHOLAR and ISO systems generally. This list for the most part merely enumerates capabilities discussed in the previous section.

(1) SCHOLAR offers mixed-initiative dialogue. The system both asks questions of the student and answers questions from the student on a given subject matter.

(2) The interaction between student and SCHOLAR takes place in a rather comfortable subset of English, but English text is not stored in the computer as such.

(3) SCHOLAR has three different modes of interaction: mixed-initiative, testing, and question-answering. Other modes (like a tutorial one for presenting new material) are simple extensions.

(4) In storing information the teacher need not specifically anticipate all the answers a student may give or all the questions answered on the basis of the semantic network.

(5) In storing information, the teacher need not specify detailed branchings after student errors, because SCHOLAR has general strategies (independent of content) for responding appropriately to different classes of errors.

(6) SCHOLAR can answer three general types of questions: (a) Wh - questions (what, which, when, where, how many, etc.) (b) Fill-in-blank questions, and (c) True-false questions. In generating questions it can produce these three types plus multiple-choice questions.

(7) SCHOLAR is responsive to the individual student to a certain degree, in that errors or questions by the student will affect SCHOLAR's subsequent questions or presentations.

(8) In response to a student's question, SCHOLAR presents only the most relevant information (based on an index of relevance stored for each piece of information). If the student would like more information on any topic, he can ask for more.

(9) The teacher has the option of controlling many variables in SCHOLAR in order to affect the behavior of SCHOLAR in very general ways. For instance, he can block the program from asking multiple-choice questions, or he can control how much time to spend on any topic after a student makes an error.

(10) The data base is modular in three levels in order to minimize difficulties in changing to different subject matters.

(11) SCHOLAR is modelled on the human tutor, and should be extendable in a way that gives it many of the same capabilities for individualized instruction that the tutor has.

(12) SCHOLAR, because of its mixed-initiative capabilities can be modified to perform other tasks. For instance if the data base stored information about a particular computer system, SCHOLAR could serve as an on-line decision-aid for system users.

III

TECHNICAL DISCUSSION

In this section we present some relevant technical problems in CAI, their role in the development of ISO systems in general, and of SCHOLAR in particular.

III.1 Semantic Networks

We will first discuss semantic networks and their general characteristics. A more precise description of the specific characteristics of the semantic network utilized by SCHOLAR will be presented in Section III.2 below. As indicated in Section I, semantic networks stem from the pioneering work of Quillian (2) in natural-language comprehension.

Semantics is the science of meaning. In linguistics, semantics is concerned with the deep structure of sentences (Chomsky, 12) i.e., with what the words and their modifiers stand for, and how different words affect each other at that level; on the other hand, the way they are organized sequentially within a sentence is in the domain of syntax. A semantic information structure is an organization of units of information in terms of their meaning and mutual relationships. When each unit in the set may refer to other units within the set, which in turn refer to other units in the set, and so on, with the possibility of loops and cross-references, we have a semantic information network.

Figure 2 was presented in Section I as a pictorial representation of a portion of the semantic network of geography of South America. Figure 4 represents a fragment of the unit corresponding to "Argentina" plus the unit corresponding to "latitude," both taken from the same semantic network.

Units are the basic components of semantic networks, and may be thought of as pieces of information to which we usually associate a name. However, there is no one-to-one correspondence between units and names, since some units have no single word as a name (like the concept common to the adjectives political, economical, social, cultural), and some have several (synonyms). Each unit in the semantic information network is essentially composed of semantic information about the unit, in the form of a set of properties. In SCHOLAR, the first element of each property is the name of the property (attribute), the second element is a set of tags used by the executive program, and the rest is the value of the property. A value can either be a set of properties

```
(RPAQQ LATITUDE (((CN LATITUDE)
      (DET THE DEF 2))
    NIL
    (SUPERC NIL (DISTANCE NIL ANGULAR (FROM NIL
            EQUATOR)))
    (SUPERP (I 2)
      LOCATION)
    (VALUE (I 2)
      (RANGE NIL -90 90))
    (UNIT (I 2)
      DEGREES)))
```

```
(RPAQQ ARGENTINA (((XN ARGENTINA)
      (DET NIL DEF 2))
    NIL
    (SUPERC NIL COUNTRY)
    (SUPERP (I 6)
      SOUTH\AMERICA)
    (AREA (I 2)
      (APPROX NIL \ 1200000))
    (LOCATION NIL SOUTH\AMERICA (LATITUDE (I 2)
        (RANGE NIL -22 -55))
      (LONGITUDE (I 4)
        (RANGE NIL -57 -71))
      (BORDERING\COUNTRIES (I 1)
        (NORTHERN (I 1)
          BOLIVIA PARAGUAY)
        (EASTERN (I 1)
          (($L BRAZIL URUGUA
            NIL
            (BOUNDARY NIL URUGUAY\RIVER)))
```

```
    (CAPITAL (I 1)
      BUENOS\AIRES)
    (CITIES (I 3)
      (PRINCIPAL NIL ($L BUENOS\AIRES CORDOBA ROSARIO
            MENDOZA LA\PLATA TUCUMAN)))
    (TOPOGRAPHY (I 1)
      VARIED
      (MOUNTAIN\CHAINS NIL (PRINCIPAL NIL ANDES
            (LOCATION NIL (BOUNDARY NIL (WITH NIL
                  CHILE)))
          (ALTITUDE NIL (HIGHEST NIL ACONCAGUA
                (APPROX NIL 22000))))
        (SIERRAS NIL (LOCATION NIL ($L CORDOBA
                BUENOS\AIRES))))
      (PLAINS NIL (FERTILE NIL USUALLY)
        (($L EASTERN CENTRAL)
          NIL PAMPA)
        (NORTHERN NIL CHACO)))
```

Fig. 4 The Units for Latitude and Argentina (Fragments) in SCHOLAR or a pointer to a unit (or a set of units) modified by other properties. This allows multiple embedding (indeed to an indefinite depth). In Figure 4, properties are delimited by sets of parentheses. Special symbols, like $L, are used to indicate that what follows is a list of pointers to other units.

Through its different properties and their constituents (attributes and values), each unit points to other units; "Argentina" points to "latitude" since the latter is the attribute of a property of the former. The entry "latitude" in "Argentina" points to all the information about latitude stored elsewhere under the unit "latitude." Similarly, having "Buenos Aires" as the value of the property "capital" of Argentina makes Argentina point to all the properties of Buenos Aires, its capital. When needed, properties of latitude or Buenos Aires can be transferred to Argentina. This avoids unnecessary repetitions since practically all information is stored only once. Another way of seeing this is to say that the nodes in the computer representation of the information structure are of two kinds, which, following Quillian (2) we will label <u>type nodes and token nodes</u>.

In our case, a type node is a unit pointing to an informational, multi-level list. Words referring to other nodes in the body of the unit are token nodes; each one represents a pointer to the corresponding type node, (i.e., the unit with that word as a name). By using type and token nodes, information is not unnecessarily duplicated, since it is stored only once, at the type node. Of course, this type of information structure is recursive and leads to circularities which do not represent an important difficulty and are not necessarily undesirable per se.

The transfer of properties described above is made specially evident in the case of the properties which we have labelled <u>superc</u> (for superconcept) and <u>superp</u> (for superpart). The superconcept of a unit is another unit of which the given unit is a part. (Note that Quillian uses the word "superset" for what we have called superconcept.) Properties of the superconcept are directly transferred to the unit, unless specifically modified in it. When we say that a battleship is a warship and that a warship is a ship, all properties of warship, and through it of ship, apply to battleship. In the case of Argentina, the superconcept is country and the superpart is South America; the latter allows some inferences with respect to values of properties like area, temperature range, population, language, etc.

Units connected as described above form a complex network of facts, concepts, and sometimes procedures; the latter have (for the first time, we believe) been mixed in SCHOLAR with descriptive information. They are either function calls or LISP lambda-expressions, and are only distinguishable through a special flag. An example of a procedure within the information structure is that for inferring the climate of a place given certain local conditions like latitude, altitude, etc. In other words, if the climate of a place is not given factually (in terms of temperature, precipitation, etc.), it can be inferred with good probability of success knowing the latitude, altitude, etc. A detailed description of the characteristics of the network used by SCHOLAR will be presented in Appendix A, Section A2.

III.2 Relevancy and Context

If we are going to let a program like SCHOLAR carry on its own a mixed-initiative dialogue with a student with no anticipation of the details of that dialogue, we must give such a program the capability to operationally deal with the concepts of relevancy and context. Quillian(3) has appropriately said that in a semantic network, the meaning of a word, phrase, sentence or event is the whole network as seen through it. As one moves away from this point of origin though, the information in the network becomes less relevant. The notion of contextual relevancy is all important for maintaining continuity and meaningfulness in the dialogue, by asking contextually relevant questions, and by answering student's questions with relevant information and not everything that could possibly be said about the questioned matter. In this last respect, suppose the student asks, for example: What is Montevideo? Then we would like to say that it is a city, the capital of Uruguay, and perhaps give its population size, but not details like the average precipitation in Montevideo during the month of January.

We would like to have a metric to define the relevancy of a property or fact in terms of a given concept. It turns out that it is easier to establish a metric for irrelevancy, which could be defined in terms of the distance in a graph-theoretic sense from one node to another in the semantic network. Then we can operationally say that all elements within a given distance of a node are within the context of that node. That maximum distance thus acts as a threshold of relevancy.

The graph-theoretic sense, however, does not seem to us to be refined enough to be capable of handling all necessary cases. For example, it would not discriminate between two equally deep properties, one subjectively important, the other less so. For example, the latitude of a city seems subjectively more important or relevant than its longitude. At the same time, it seems natural to formally put those two properties in parallel, which implies the same formal depth. The solution to this apparent paradox can be obtained through tagging the properties in order to modify their relevancy (or irrelevancy) without changing their positions within a unit. Those tags must be assigned by the person constructing the semantic network (though some generalizations could be automated) and end up being an expression of his judgment on the relative importance of different items.

The "subjectivity" of the network evidenced by tagging is not an artifact. Two equally knowledgeable teachers would create semantic networks with slightly different configurations when dealing with the same subject matter. This is not an exclusive characteristic of ISO CAI. In AFO CAI, the same two teachers would create two different sets of questions. And for that matter, they would have different behavior in a classroom.

III.3 Natural-Language Man-Computer Communication

In SCHOLAR, we have been able to achieve a large degree of freedom in communicating with students—better than our early hopes, both in input and output (i.e., in comprehension and in generation). This has been obtained by taking a pragmatic approach which has proved successful. First of all, and, instead of attempting to comprehend all classes of input, we have restricted student answers to SCHOLAR questions to certain types, namely numerical atomic, and lists of atoms, though other elements like auxiliary words can also appear. The underlying reason has not been difficulties in parsing complete sentences, but judging their acceptability as answers. The above limitation has represented a trade-off, since as a consequence of it we had to be more demanding in the generation of questions in order to produce expected answers only of the types mentioned above. We now feel confident that an extension to simple complete sentences will be possible in a future version of SCHOLAR.

The case where we have allowed complete sentences with a large degree of freedom is that of questions asked by the student to SCHOLAR. For questions, a thorough study shows that a taxonomy can be established which facilitates their comprehension (see below). In any case, we have seen the difficulties of investigators who want to deal with all possible cases, even those very complex and unusual. We have decided to leave those special cases aside and concentrate on methods to solve most practical ones. When SCHOLAR cannot comprehend a student's question, it tells the student that and asks him to rephrase the question; if words unknown to SCHOLAR appear, it points them out. This is, after all, what a human would do.

A similar approach has guided the generation of English sentences by the computer. The strategy has been to use short sentences with no embedded clauses and a limited repertoire of verbs (see Appendix A). This has proven to be highly successful.

Finally, let us emphasize that most programs producing acceptably constructed English output do so at an elementary technique of replacement within formats, like Weizenbaum's ELIZA (Weizenbaum, 13) does. SCHOLAR is more creative since all sentences and questions generated by it involve a complete processing, from a semantic internal representation into English.

III.4 Questions: Their Nature and a Possible Taxonomy

In SCHOLAR, questions are asked to and by the system. In neither case are questions and answers textually stored; in the former case, questions must be interpreted in terms of the data base; in the latter, questions must be generated from that data base. It is, therefore, necessary for us to have good understanding of what questions are and what types exist. Surprisingly, we have found very little in the literature that could be utilized in a practical sense to help us in this task.

There are clearly two aspects of each question: the semantic aspect (i.e., what the question is about) and the syntactic aspect (i.e., how the question is formulated). For example, the question "How many people are there in Brazil"? refers to the string "70000000 population Brazil" and tells us specifically that in this string we are questioning about the value 70000000. In terms of the form of the question, we see that it is a "how many" question which perhaps could be considered as a particular case of a "WH" (i.e., what, which, where, who, etc.) question. In Appendix A, we describe how SCHOLAR can "understand" such a question.

All questions involve information retrieval in one form or another. Most simple questions in a subject like geography are direct requests for retrieval of certain information. We will call object the item which is the object of the question, i.e., the concept being talked about. We will call attribute the aspect of that object we want to know about; value is the information obtained when the attribute is applied to the object.

There is a convenient correspondence between object-attribute-value triples and our semantic network. In a simple case, Montevideo (value) is the capital (attribute) of Uruguay (object). We see that here we can use the unit of information on Uruguay, in which we find a property named capital, with value Montevideo. It is clear that more complex objects, attributes, and values can exist. The value can clearly be a complex tree. The object can also be complex through recursion and/or conjunction or disjunction. The same applies to the attribute. We can thus have questions like:

> What is the average summer temperature in the capital of Uruguay?
>
> Give me the latitude of Montevideo and the population size of Brazilia.
>
> What are both the climate and the area of either Uruguay or Chile?

In the first example above, we cross the limits of a unit. We retrieve Montevideo as the capital of Uruguay in the unit "Uruguay," and then find the average summer temperature in the unit "Montevideo." The formalism of using triples for questions also clarifies a final semantic taxonomy proposed by several authors (Rovner and Feldman [14], Johnson, [15]) in relation to storage schemes based on triples, though storage in terms of triples is not required in order to apply it. This taxonomy is based on what element is being questioned in a triple. We can have different cases. Let us consider questions based on the simple string: Montevideo capital Uruguay. By questioning each one of the three elements, the following questions are originated:

> What is the capital of Uruguay? (Value)
>
> Montevideo is the . . . of Uruguay. (Attribute)
>
> Of what country is Montevideo the capital? (Object)

The most common case is the first of the three above. If both attribute and value are missing (and being questioned) we have:

> Tell me about Uruguay.

If none of the three elements is missing, we have an assertion, which can be related to a true-false or yes-no question. For example:

> Is it true that Montevideo is the capital of Uruguay?

Of course, the true-false question may be a false statement which generally implies replacing the value by another expression usually of the same kind (i.e., same superconcept). The following question is formed that way:

> Tell me if the capital of Uruguay is Santiago.

An extension of the generalized true-false question is the multiple-choice question, in which different alternatives (usually four or five) are presented either to complete a sentence by filling one or more blanks or to answer a WH-type question (a WH-type question is one containing "what," "where," "which," etc.). Only one alternative is correct.

From a syntactic point of view, we distinguish the following types of questions:

a) Yes-no, and true-false questions. In these questions there is only a binary choice in the response.

b) Multiple-choice questions, discussed above. The choice is not binary, but it still refers to a closed set of alternatives, the correct one plus several wrong ones that must be generated, and the response is not constructed. Multiple-choice questions usually use plausible alternatives but sometimes some unreasonable ones are used (like a negative number for an area or a population size). Multiple choice questions involve a question of type (c) or (d) below. Finally, many teachers dislike multiple-choice questions, mainly because in certain subjects wrong associations may develop when students are exposed to wrong alternatives.

c) WH-type questions. These questions involve a constructed response, through the use of words like "what," "which," "where," "when" and other interrogative words. We will also include here imperative sentences with commands like "tell me about," "give me," "name," etc.

d) Fill-in questions, which consist of an assertion in which a missing portion must be supplied by the student. (c) and (d) are essentially equivalent with minor differences.

e) Imperative statements leading into the application of sine procedure, through commands like "compare," "conjugate," and "translate." Observe that technically these are not questions, but can be treated as such.

f) Some special types of questions which might be desirable in certain subject areas. One of these is, for example, the transformation type used in language courses, in which a sentence or paragraph must be converted from present to past, or from singular to plural, or affirmative to negative.

g) The essay question, in which the respondent freely constructs a fairly extensive discourse on a topic specified in the question. In SCHOLAR we will admit some requests of this type by the student, with little essays constructed by SCHOLAR (see Appendix A). Essay-type responses made by the student are very difficult to process. This is a major research problem yet unsolved, and will not be considered any further.

SCHOLAR can at present comfortably handle question-types (a) through (d), and (e) to the extent in which the related procedures are available. These capabilities of SCHOLAR apply both to questions asked by it and to it.

SCHOLAR can also handle another important aspect of questions, which we have not discussed so far. This aspect is quantification through modifiers like "some," "all," "everything," "something," "more," "most." These can appear in questions by SCHOLAR as well as in questions asked to it. The latter is the most interesting case. Thus, the unquantified question

What is the topography of Argentina?

can yield the following quantified ones:

I want to know something about the topography of Argentina.

Tell me more about the topography of Argentina.

Tell me all about the topography of Argentina.

Incidentally, SCHOLAR handles the answers by using its capabilities to assess contextual relevancy (see Sections III.2 and A.4). There is also possibility of quantification in terms of the elements of a list. For example, we could generate or answer the following questions:

What are the countries in South America?

Name some countries in South America.

Name most countries in South America.

Give me three countries in South America.

What are all the countries in South America.

We must finally say, before ending this Section on questions, that even after the basic selection of a question from both a semantic and modal point of view, there is room for some stylistic variations. For example, SCHOLAR will "understand" and answer any of the (essentially equivalent) questions:

Is Montevideo the capital of Uruguay?

Is the capital of Uruguay, Motevideo?

Tell me if Montevideo is the capital of Uruguay.

Tell me if the capital of Uruguay is Montevideo.

Is it true that Montevideo is the capital of Uruguay?

Is it correct to say that the capital of Uruguay is Montevideo?

III.5 Error Detection, Diagnosis, and Remedial Action

The whole area of error analysis, diagnosis and consequent remedial action is one of the most promising avenues open to ISO CAI systems, in opposition to classical AFO CAI systems. In classical AFO CAI, there is usually full anticipation of correct answers as well as certain incorrect ones with their corresponding branching. Either there is no possibility of unanticipated answers (because of restricting questions to closed-set, multiple-choice ones) or there is a category for all unanticipated answers, with a pre-established consequent action.

In ISO systems, while generating a question, we can at the same time generate a correct answer (or a set of them). It seems natural to use that derived correct answer as a standard for matching with the student's answer. This is convenient and is the strategy adopted in SCHOLAR. For this strategy to be effective, it is necessary to have a more or less unique and well-defined correct answer, or well-defined closed set of correct answers (a list of items, for example). On the other hand, free complex constructed responses really represent open sets and cannot be checked by matching techniques. As said before, in the present version of SCHOLAR we have designed our question-generation routines in such a way that expected answers are either atoms, lists of atoms, or numbers.

One must mention, of course, that an answer may differ from the expected answer, and still be acceptable and considered correct. In SCHOLAR, there are three cases of the above; the program is set to accept misspelled words if they are "close enough" to the expected words; it is also set to accept synonyms; finally, on numerical answers, the program can accept numbers approximately equal to those expected. These features can be blocked by means of suitable parameters. (See Section A.9 below).

Let us also add that an answer may not be correct or wrong in absolute terms. For example, a question like "What are the countries in South America?" can be answered with a list of only nine of them, or with most of them plus one Central-American country. In this case, the error-analysis procedures must separate the correct and wrong parts of the answer as is done in SCHOLAR (again see Section A.9).

Beyond detecting errors, one would like to classify them, take proper actions to correct them, and understand the reasons for those errors to occur. In a sense, errors can be considered as the symptoms of diseases which are the reasons for their occurrences. We are faced, then, with a diagnostic task but one which should operate on an open set of alternatives.

It is clear that AFO systems have little or no capability to perform diagnosis by themselves. They can only follow specifically the directions left by the teacher as a result of his possible "pre-diagnostic" efforts on predictable answers. Only ISO programs, possessing a data base organized on the basis of knowledge about the subject matter, will have the potential capability for probing that data base and utilizing it to find reasons for an unanticipated, observed student error. SCHOLAR is a program which has been constructed with the basic conditions to eventually have some good diagnostic capabilities; in the balance of priorities, though, we have concentrated so far in developing those basic conditions in SCHOLAR , and only to a minor extent, in the development of its diagnostic capabilities. The limited capability implemented now can recognize some classes of errors and generate related questions. Usually, however, the correct answer is given when an error is found.

III.6 The Teacher Interactions in CAI

The teacher-computer interaction is usually necessary at three levels:

(1) Preparation of the data base, be it in the form of questions and frames in AFO CAI, or a semantic network in ISO CAI.

(2) Setting conditions for student-computer interaction, i.e. defining the system parameters necessary to stimulate the conditions of that interaction.

(3) Collection of results, in the form of scores, statistics, and general history of the student-computer interaction after it has taken place.

There is a possible fourth role for the teacher in CAI. This role is that of a supervisor in real time of the actual operation of a system with many terminals. When, for example, a system like SCHOLAR is asked a question for which it has no answer, instead of answering something like "Sorry, I don't know," (as it now does), it would ask for help to the human supervisor. The system could also ask for help in the case of complicated diagnoses, etc.

In SCHOLAR we have concentrated our efforts on the student-computer interaction; priorities have forced postponement of the programming of most teacher-computer interactions. A small conversational program to help the teacher set the student-computer interaction conditions exists, and will be described in Appendix A. So far no implementation exists, however, for a teacher program of type (1). The problem is now more important and interesting than before, since we are faced with the construction of a semantic network. In spite of the lack of implementation, we have given this problem considerable thought. Two possible approaches can be attempted.

The first approach follows the line of studies on natural-language comprehension (Quillian, 3). This consists of reading English text into the machine; the program attempts to code that text in the format chosen for the internal structure. The program checks with the person entering the material to see if it has been properly coded, and calls his attention to undefined terms, and other anomalies.

The second approach, which we advocate for ISO CAI data bases, is one step more conservative than the first. It involves building a conversational program to help the teacher encode the material in an incremental way, saving parsing and other problems. Heavy use can be made of semantically higher levels (through superconcept and other superordinates) if they have been already coded.

Figure 5 illustrates a simulated teacher-computer protocol. In the protocol, the computer uses its knowledge about the concept country and its properties to help the teacher encode the unit Chile. In the case of Cordoba, the computer does not know what a province is and proposes to define that first. If the teacher overrules this, he would still be able to encode properties about Cordoba.

The teacher should be able to specify other properties about the location of Chile that are not specifically asked for. This he would do by answering yes to the question "More about the location of Chile?" Then he would be asked about the name of the property and the value as happens below the line in Fig. 5a. (The teacher should always be able to say no, such as to the question about what continent Chile is in.) The teacher should be able to interrupt later if he remembers any other properties he wanted to enter about the location.

To a certain extent, the teacher will have to adapt his coding to the form of the data structure, but he should be able to enter any property that he wants to. To the degree he has to conform to the data structure, he will have to learn the conventions of the system; but they are not many. The major ones can be seen in the example: breaking down of properties into name and value pairs; specifying the part of speech into one of ten categories; and assigning values to tags that indicate the relevance of each property to the unit, in this case Chile. The first of these is the only one that could cause difficulties.

Of the three levels of the data base to be discussed in detail in Section III.7, instructional programmers will only have to worry about the lowest level (which represents the major portion in length).

Name of unit:

Chile

Part of speech:

XN

Superconcept, please:

contry

Do you mean country? Answer yes or no.

Yes

In what continent is Chile?

South America

More about the location of Chile?

No

Area of Chile in square miles:

If you want to add some properties, type first the name of the old property, if any, in which they are included; if none, type none.

Population

Name of new property:

Language

Value

Spanish

Property

Done

Fig. 5 (a)

Simulated Protocol
of a Teacher Building the
Semantic Network.
(CONT.)

Names of properties and subproperties you want to tag for irrelevancy:

Location, area, population

OK. Location:

2

Name of unit:

Cordoba

Part of speech:

XN

Superconcept, please:

Province

Undefined. Do you want to define it now?

Yes

Part of speech:

CN

Superconcept, please

Fig. 5 (b)

Simulated Protocol
of a Teacher Building the
Semantic Network.
(CONCLUDED)

General concepts about the field in question (such as latitude, etc.) would have been already coded by specialists, and general English terms (the highest of the three levels of the data base) are a part of the SCHOLAR system itself, together with the general programs.

III.7 Overall Structure of SCHOLAR

Figure 6 illustrates the general structure of SCHOLAR. The student, represented by a program acting upon an information structure, has SCHOLAR as a counterpart, also with a program and information structure as main components. The information structure in SCHOLAR centers around the semantic network of facts, concepts, and procedures. We have attempted to modularize the semantic network into three levels:

(1) a general level which is context-independent and which contains information about English words and concepts necessary no matter what the applied subject matter is. Here we have items like: prepositions; general verbs like have, be, do; interrogative and negative words; modifiers and quantifiers like approximately, usually, very, some, all, a few; other adjectives like large and varied; determiners; pronouns; etc.

(2) An applied level which contains general information about the area of application, in our case, geography, but not about particular examples. Here we include units like climate, country, temperature, hot, temperate, degrees Fahrenheit, etc. Problem-dependent procedures like that for climate which we discussed earlier, also go here.

(3) The level connected to the specific context that serves as an example, in our case, South America. Here we have mostly what we have called example nouns (XN) though sometimes we may have some adjectives. Examples are Paraguay, Paraguayan, Aconcagua, Brazil, South America, etc.

Within the units at any one level there are of course references to concepts stored at the other levels. For example, "Argentina" at the specific context level refers to "latitude" at the applied level. Generally, these references are only from a more specific

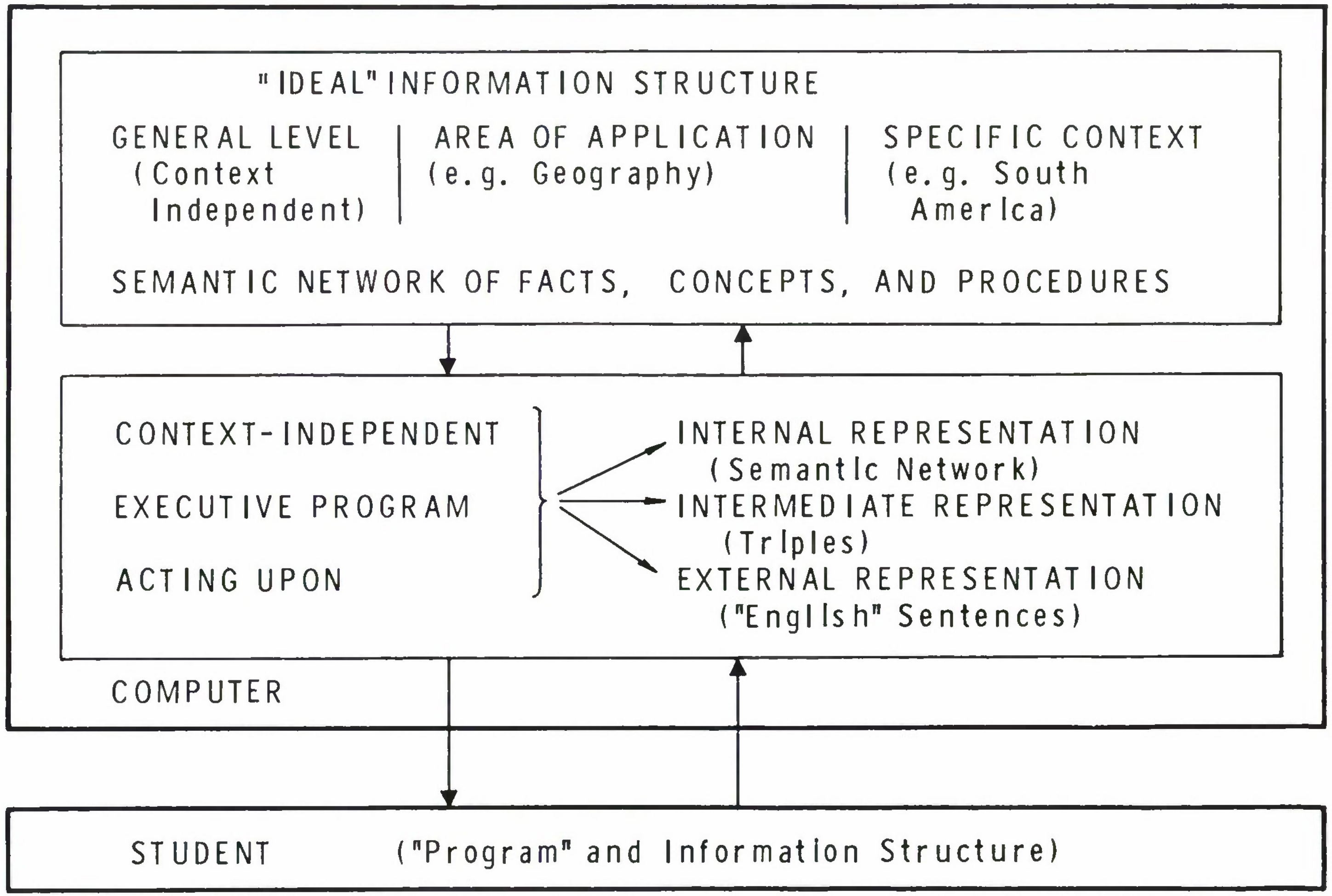

Fig. 6 General Structure of SCHOLAR

level to a more general level, though there are some references in the other direction as mentioned below.

Due to this modular construction, it is possible to replace the specific context by another context in SCHOLAR's level (3) without any major effect on levels (1) and (2). This way we can replace South America by New England, or the Middle East, without substantially modifying general linguistic or geographic information. Similarly, we could modify both levels (2) and (3), going to, say, anatomy of the circulatory system, without any major revision either in (1) or in the executive program.

Some minor adjustments will be necessary in a high level when a lower level is replaced. These revisions have two causes. First, we have found convenient to have some redundancy in the network, with some pointers back, in order to facilitate certain searches and associations. For example, units on individual countries have pointers to "country," but it is also convenient to point back from the unit "country" to individual countries through a property which may be called "examples." Some of those back-pointers will have to be modified when we change the specific context from South America to something else.

The second reason for a slightly imperfect modularity stems from the nature of some properties. For example, we may want to change our working definition of large in terms of the area of a country when we go from South America to Central America (the latter with much smaller countries than the former).

Apart from the semantic network and some other data to be discussed in the following subsection, SCHOLAR includes a context-independent executive program. Being context-independent means that changes in the semantic network will not require modifications in the program. This is another consequence of the modularity considerations permeating the design and implementation of SCHOLAR.

As shown in Fig. 6, the executive program acts upon data at three levels. First, it processes information in terms of its internal representation in the semantic network. At the other end, it handles input and output in a subset of English through a package of input-output routines; this subset of English is, we reiterate, rather ample and unconstrained. Finally, and in order to act as a bridge between the internal representation in the semantic network and the external communication in English, we have designed an intermediate representation in terms of object-attribute-value triples (for example, "Argentina" might be the object, "capital" the attribute and "Buenos Aires" the value). This intermediate representation is a convenient break to facilitate the conversion between external and internal representation. It is also especially convenient to express retrieval requests, but triples are not the internal representation itself.

III.8 Operation of SCHOLAR and Its Agenda

As we mentioned in the Introduction, there are three modes of interaction with the student that have been programmed. These are: the mixed-initiative mode (mixinit) in which either side can ask questions in a dialogue form; the testing mode (test) in which only the computer can ask questions, rejecting those by the student (otherwise this mode is the same as mixinit); and the question-answering mode (Q/A) in which the computer responds to the student's questions but asks none itself. If no mode is specified when SCHOLAR is called, SCHOLAR will operate in the mixed-initiative mode. Another mode can be requested by typing it as an argument in the initial call.

There is another pair of arguments that can be specified when the program is called; these two relate to the operation of the agenda in SCHOLAR. The agenda determines what context (or topic) is to be discussed and how long to spend on each context. These are the two arguments that can be specified. For example, the program call could specify that "Argentina" will be discussed for "1/2 hr." (or alternatively for 10 questions). This could be extended so that the teacher (or student) can specify a whole list of contexts and times. If not specified the two arguments will be "South America" and "1 hr." This agenda is only operable in mixed-initiative and test mode where the computer controls the dialogue; in question-answering mode any topic may be raised by the student.

The agenda operates as a push down list. If the context is "South America," it is necessary for SCHOLAR to generate a subcontext e.g. "Argentina." This would be generated on a random basis (unless of course the teacher or student specifies a detailed agenda when SCHOLAR is called). When a subcontext is generated a portion of the total context time is allocated to the subcontext. Where the subcontext is still too general, as say "Argentina," then a sub-subcontext is generated in a similar fashion, e.g. "topography of Argentina." The important point is that all subcontexts are generated by SCHOLAR itself automatically, except to the degree the teacher or student wants to specify just what material should be covered.

In mixed-initiative or test mode, once a subcontext is chosen, then questions must be generated. This is done by first selecting a semantic string, then selecting a mode for the question, and finally coding it in English. The string chosen might be the attribute-value pair, "capital" "Buenos Aires." Then a question mode must be generated, e.g. true-false mode. In this case half the time the false question will be generated. This is done by generating a false answer on a weighted-random basis. For example another city in Argentina such as Cordoba is highly likely. Then the question itself is generated by another routine "The capital in Argentina is Cordoba. Correct or Incorrect?" At the same time the right answer has been identified (in this case "Incorrect") and will be used as a standard for comparison in the answer analysis phase.

When the student responds, his answer is taken in and compared to the expected answer. As mentioned the kinds of answers allowed are atoms, lists, and numbers. The comparison routines evaluate whether the student answer is the correct answer with allowance for approximate answers (with numbers), misspellings, and synonyms. Depending on the outcome of this comparison, an appropriate message is printed and a subsequent action is decided upon.

The subsequent action will usually be to check the agenda to see if time is exhausted on the current context. If it is,a new context would be generated; if not, another question would be generated in the current context. There are three other subsequent actions that may occur: (1) a temporary subcontext may be generated, when certain classes of errors are diagnosed, (2) the last question may be reformulated as a simpler true-false question, and (3) SCHOLAR may ask the student to try again.

If the English interpreter detects that the student's input is a question, or a question-like command, it will call the question-answering (Q/A) module. After reading and interpreting the input, retrieval and/or other information-processing routines produce an answer, which is then converted into English sentences by the English-text generator. In Q/A mode the program would then loop to accept a new question. In mixed initiative, however, the question-answering routines return after a single question. Upon return, the student is prompted to give the answer to SCHOLAR's former question, and we are back in the previous track.

If at any time the English interpreter recognizes the student input to be the name of a general interaction mode (Q/A, etc.), it asks the student if he wants to change the mode of the interaction. Upon an affirmative answer, SCHOLAR changes the mode of the interaction to the requested one, and proceeds in this new mode. The student can also ask to terminate the interaction by typing EXIT or WRAPUP. Though not fully implemented in the current version, the WRAPUP type of exit is designed to present the student with a list of topics in which he needs further work.

The student also has the option to type at any time, even when it is not his turn to type, the symbol "#". SCHOLAR periodically checks the input buffer, and if it contains that symbol, an interrupt occurs, similar to that described above when the student typed a general interaction mode.

IV.

CONCLUSIONS

IV.1 Some Implementation Considerations

SCHOLAR has been implemented in BBN-LISP in an XDS-940 time-sharing computer. Conversion to a larger and faster Digital Equipment Corporation PDP-10 with hardware paging is under way.

BBN-LISP (Bobrow et al., 16) is a sophisticated and versatile version of LISP. It was the first paged version of LISP available and very successful thanks to skillful heuristics followed in space allocation (Bobrow et al. 17). The fact that the system is paged (which is invisible to the user) gives the user a virtual memory considerably larger than core memory; this was the major consideration guiding the selection of this environment for the implementation of SCHOLAR. Another feature of BBN-LISP is its excellent conversational editing and debugging capability; this is very convenient when developing a prototype system of the complexity and size of SCHOLAR.

The fact that SCHOLAR is being developed in LISP does not mean that all ISO systems, in particular practical systems of the future, will need to be coded in LISP. BBN-LISP, because of its flexibility and its list-processing capabilities, is particularly good for developing prototype systems of this type. When the implementation problems have been solved, however, it will be possible to reproduce the system in other languages including assembly languages. We want to emphasize here that there is little about the way SCHOLAR works that makes using LISP preferable beyond the developmental phase.

Let us now give some statistics on SCHOLAR. It essentially takes all available space give by the BBN-LISP system on the XDS-940, which is 144K (K=1024) 24-bit words. Each LISP word takes two XDS-940 words, while binary-program, compiled-code, and array words take one. After taking some auxiliary portions out, we have some 35K occupied by the LISP system. Next, SCHOLAR (program and data) takes on the order of 45 x 2K words most of which is program lists (the program is running interpretively, see below). The semantic network is approximately 6K LISP words, i.e. 12K 940 words. Space for other data is roughly equivalent. The space taken by names is not included above; in SCHOLAR it is around 10K 940 words. The rest, of the order of 10K 940 words, is the working space with which the program operates (temporary lists and atoms).

In terms of speed, the XDS-940 time-sharing system is quite sensitive to the presence of many users, particularly those with large programs which force heavy paging. With a very light load, answering a student question now takes approximately one minute. This figure is deceiving, however, because by our own choice the program is currently running interpretively. The reason for this is that, since SCHOLAR is an experimental system and not a final product, changes and experiments are constantly being made; having compiled code would have forced very frequent recompilations. These are particularly inconvenient because limitations of space prevent the possibility of having the interpretive version around when running the compiled one; necessity for frequent reloadings is the consequence. Experiments done with compiled versions, however, follow the general results obtained in BBN-LISP. These indicate that approximately a fifteen-fold increase in speed is obtained by compilation. The next factor will appear when conversion to the PDP-10 is made. Very conservative estimates would yield here a factor of four as gain in speed. Combining both factors we have a conservative estimate of a sixty-fold gain in speed for a compiled version on the PDP-10. This would bring the response time to a student question down to approximately one second, a very reasonable figure indeed.

As we said in Section II.2 ISO CAI seems to require for its implementation a fairly large system in terms of memory. This suggests that a suitable environment for it can be a large time-sharing system for many users rather than the small computer with a single user. On the other hand, though we could have a time-sharing system dedicated to SCHOLAR serving many students, this is not a requirement. SCHOLAR can coexist, sequentially or simultaneously, with many other programs in a large, multi purpose time-sharing system. An example of such an environment is provided by the versatile TENEX system at BBN which can accomodate many users with widely different computational requirements.

It seems appropriate to include in this subsection a look at what features we would like computer systems to have in order to help the future development of ISO CAI systems.

One of them is clearly larger memories. Paging allowing large virtual memories at the expense of a loss in speed may not be the ideal long-term solution. We would like to have much larger direct-access memories. A moment of thought indicates, however, that we would like those larger memories to store much larger and intricate semantic networks. Therefore, we could safely assume that, after being built, those networks need not change during students' interactions. The use of read-only fast optical memories is a suggestive possibility in that respect; it may be practical in the not-too-distant future.

Parallel-processing capabilities could be an important advantage for computer systems using semantic networks as data bases. The frequent searches fanning out from a given node would benefit considerably.

Within LISP, we would like to see the capability for having overlays, in order to replace parts of the semantic network without the need to load the new material (with the consequent garbage collections). Some problems, like the effects of pointers to structures that have disappeared, exist; hopefully, those pointers could be reduced to a minimum through some modularization.

Let us conclude this section on implementation with a note on speed, efficiency, and cost. We have already considered running speed, and shown that it could be quite satisfactory. About efficiency, it is a desirable quality, but not essential in an effort like SCHOLAR. We were not trying to build an efficient CAI system, but to demonstrate that a new type of CAI is feasible; efforts to optimize coding will come later.

Finally, what about cost? For some time ISO systems will be too expensive to be used by real students. In a not too distant future, however, more powerful ISO CAI systems will be built in computers better suited to them. These computers are presently being designed and experimentally built at the time of this writing. And, in any case, we should not wait to have those computers and then develop the scientific bases and the software technology to use them. Besides, if we do not consider the cost per lesson, but some cost as a function of learning and achievement, and we include the cost of teachers' time, it is possible that the break-even point between AFO and ISO CAI systems may occur much sooner than what more conservative and limited considerations would predict.

IV.2 General Conclusions

In this report we have presented the first prototype of a truly mixed-initiative interactive system capable of conversing in a subset of English with the user. Little or no anticipation of specific items and sequences in the conversation is required. Though we have examined this new type of system within a training environment, its relevancy in other areas of great concern to the Air Force (like command and control, intelligence and logistics) is clear. One specific new application of the basic ideas developed in SCHOLAR is that of an on-line aid in decision making based on using complex and highly structured data bases (like those in command and control). Another possible new application is that of an on-line helper in systems training for developing expertise in users of a new computer system and/or a new computer language.

Within the domain of the specific environment in which we have constructed SCHOLAR, we can say that we have proved the feasibility and shown the basic capabilities of a new kind of CAI systems which we have labelled ISO systems (for information-structure-oriented. They are an attempt to improve upon classic ad-hoc frame-oriented (AFO) systems which are based on detailed specification in advance by a human teacher of textual material, questions, correct and incorrect answers, and conditional actions. On the contrary, ISO systems require no detailed anticipation; they require instead an information structure which symbolically represents knowledge on the subject being discussed. A generative program operates on that information structure, constructs answers to student's questions, and originates appropriate questions and the corresponding correct answers. This leads to mixed-initiative man-computer dialogues in which either side can interrogate the other. The dialogues can take place in a rather unconstrained and comfortable subset of English. The value of this approach for decision aiding is apparent.

Human performance aiding systems of the ISO kind have not existed unitl now. Our task has been to prove that they can be built, and we have done this by example. The example is SCHOLAR, a prototype ISO system capable of conducting a mixed-initiative review of the knowledge of a student about the geography of South America; the construction of the system and the data base are modular, and SCHOLAR could be applied to many other topics (in geography and otherwise) with only very minor adjustments.

The detailed capabilities of SCHOLAR, and their implementations have been discussed in Section IV. Some of the modules (at different levels) represent only one possible solution, and they could be replaced without changing the basic philosophy of the approach. In a sense, SCHOLAR, more than a final product itself, is an environment in which variations in techniques and strategies can be formulated and tested.

IV.3 Recommendations for Further Work

There are many possible and necessary lines of work stemming from the research here reported. Let us briefly state the main ones:

(1) Refinements and extensions in terms of program. Branching after errors should be an important concern here. The incorporation of some additional inferential capabilities has also importance. Completion of the work partially done on providing answers to questions which are based on generalized computation (like "compare" and "conjugate") should also have high priority.

(2) Extension of the data base in terms of both content and size. In terms of content we would like to create an ISO system in some area where some additional capabilities could be tested. One of these is the presentation and monitoring of examples and problems, which do not necessarily have to be numerical. In terms of size, we would like to create a data base, say, ten times as large as that in SCHOLAR. We do not anticipate serious problems here because of the structure of our semantic network; problems would be much more serious if we had an internal representation based on more elementary units, such as triples.

(3) Design and implementation of procedures to help the inststructional programmer construct the data base. In Section III we proposed to do this in a high interactive way with the computer leading the teacher as far as possible.

(4) Investigations of discourse and teaching strategies, and related matters. We are worrying about problems like the following: Is there a reasonable working taxonomy of errors which can be considered to be content-free? What actions should be taken after each type of error? How should the specific context of the material to follow be chosen? When and to what extent should the correct answer be presented to the student? How do we select the content of a specific question, given a specific context and, perhaps, other constraints, like recent errors? Given that we have a string representing some meaning, what is the best way to form a question about it, or in other words, are there preferred question modes under well defined circumstances.

The modular nature of SCHOLAR and the fact that the program is independent of the content of the data base makes it an ideal vehicle to be used as a tool for research on discourse and teaching strategies. Furthermore, we want to claim that the design of ISO systems like SCHOLAR not only provides a good environment for research in that area, but also motivates it. Having to design an ISO system, we are forced to define and provide solutions for important questions poorly defined so far. Our point here is that SCHOLAR and similar ISO systems constitute an ideal environment for research on pedagogical questions; they are not only capable of serving as tools to provide answers, but they also force the formulation, in precise but general terms, of some questions of vital pedagogical importance.

(5) Extensions to applications outside the specific domain to which CAI has been traditionally applied (i.e. teaching of scholastic subjects), but where mixed-initiative contextual dialogues should be an important asset. This includes all types of performance aids, and, in particular, on-line help in systems training. A logical continuation of the effort reported in this document would be a mixed-initiative system available on-line to assist a novice user learning to use a new computer language or a new computer system.

APPENDIX A

THE CURRENT IMPLEMENTATION OF SCHOLAR*

A.1 Overall System Organization

In this Appendix we will discuss in detail the current implementation of SCHOLAR. The background and technical approach have been already discussed in the main body of this report. We will here present what is specific to SCHOLAR, avoiding general discussions. Whenever convenient, we will illustrate our description with diagrams or computer printouts indicative of different aspects of the program.

First we will describe the overall behavior of SCHOLAR when it interacts with a student. Here the reader may want to refer back to Fig. 1, the rather extensive on-line protocol of the conversation between SCHOLAR and a student presented in the Introduction.

Three modes of interaction with the student have been programmed. These are: the mixed-initiative mode (mixinit) in which either side can ask questions in a dialogue form; the testing mode (test) in which only the computer can ask questions, rejecting those by the student; and the question-answering mode (Q/A) in which the computer responds to the student's questions but asks none itself.

The program is called by typing "SCHOLAR ()"; there is then a brief initial interaction after which, if no mode has been specified, SCHOLAR will operate in the mixed-initiative mode. Another mode can be requested by typing it as an argument in the initial call. This is the third argument in the procedure SCHOLAR; the first gives the opportunity to call a specific name as an overall context; the second permits the optional specification of the number of questions to be asked. Normally, all three arguments are NIL. In that case SCHOLAR operates using its agenda for overall context, time for limiting the duration of the interaction, and mixed-initiative as a mode.

Figure A.1 is a schematic block-diagram of the operation of SCHOLAR. After the box labelled INITIAL which initializes the program and conducts the initial interactions, branching occurs depending on the mode. The test and mixinit modes follow a similar path, except that in the test mode student's interruptions and questions are rejected.

---

*This Appendix is considerably more technical than the rest of this report, and is intended for readers interested in a detailed discussion.

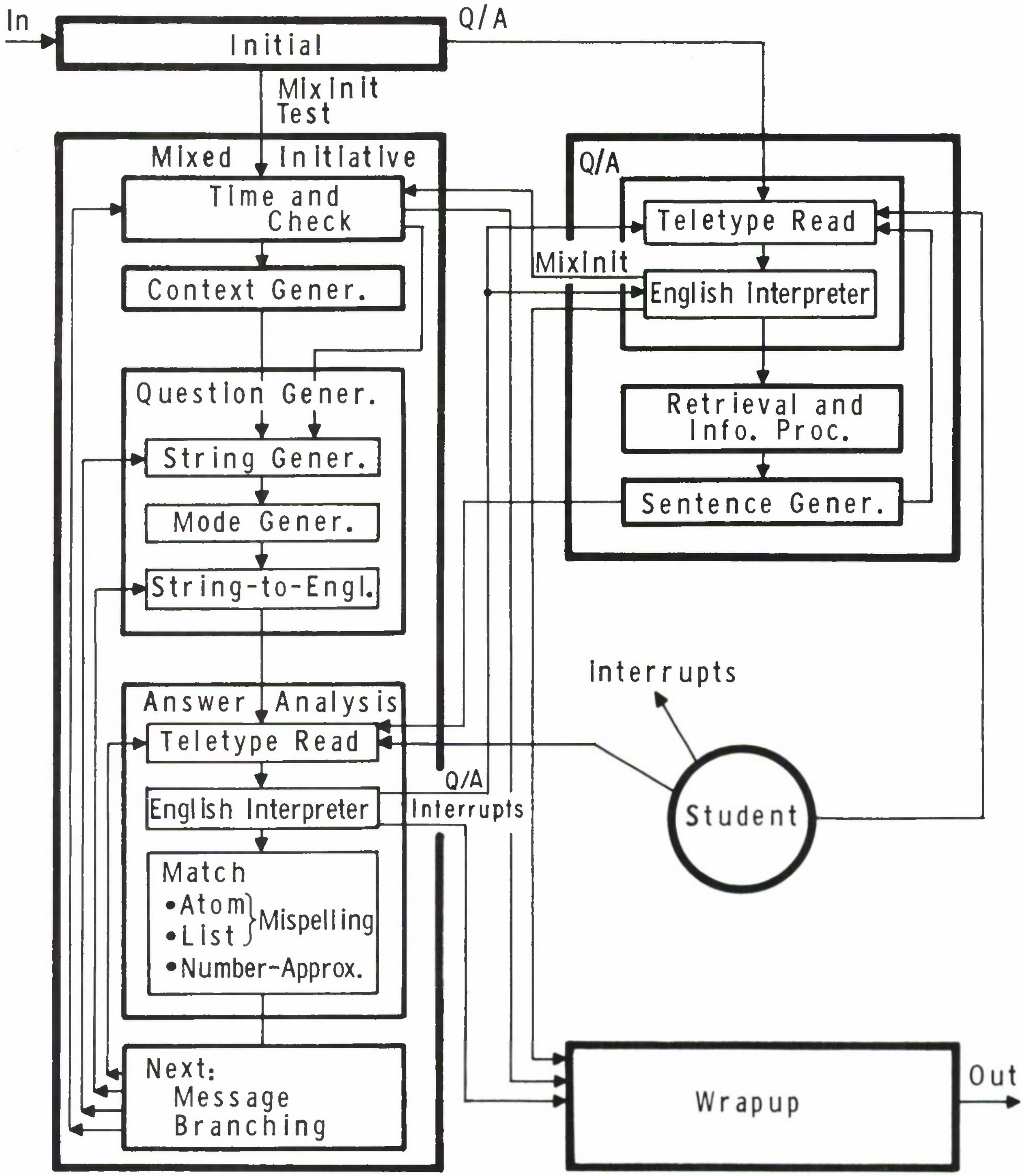

Fig. A.1 Schematic Flow-Chart of SCHOLAR

We will not consider the test mode any longer since it is really a simplification of the richer mixinit mode. Also we will here discuss only the overall structure of SCHOLAR. The more detailed operation of the different modules will be discussed in later subsections.

As the top-level procedure, SCHOLAR calls in a sequence: first INITIAL, then SCHOLAR, and finally WRAPUP (which dismisses the student). SCHOLAR, in turn, calls either MIXINIT, Q/A, or TEST, depending on the mode, and also handles changes from one mode of operation to another.

In the mixed-initiative mode, there is first a check for time and context (see below). If necessary, a new context is then generated, and in all cases, a question within the current context is formed, by first generating a semantic string, then selecting a mode for the question, and finally, coding it in English. At the same time, the correct answer has been identified and will be used as a standard for comparison in the answer analysis phase. The student answer is then processed and interpreted; next a package of matching routines compares the expected answer with the actual answer. These routines allow the processing of atoms, lists, and numerical answers, with provisions for approximate answers, misspellings, and synonyms. The procedure NEXT receives the result of the matching prints appropriate messages, and decides on subsequent actions, with possible branching. (Observe that in an ISO system the word branching looses part of its meaning since we do not have a closed set of alternatives anymore.)

The stage following NEXT can be one of the following four. The first is a repeat of the whole sequence. (Now the time and contextual checks are more meaningful than in the first pass.) Either new question on the old context, or perhaps a whole new context, followed by a question on it are generated, etc. The second alternative is to come out of NEXT with a definite context usually different from the one which SCHOLAR had been using; in this case we may enter directly into the question-generation procedure at the string-generation level. This occurs in attempts to diagnose confusions where we may want to ask questions about specific items or topics; these items or topics are added at the top of the context push-down list as temporary sub-contexts which are usually given a short life. The third alternative is to reformulate the last question in a different form. This is done in certain cases by reformulating the previous question as a true-false one. In this case the operation after NEXT is the string-to-English generation (since the same string used before is utilized again).

Finally, the fourth alternative is to give another chance to the student, i.e., the "try again" type of action. In this case and after printing an appropriate message SCHOLAR loops directly to the teletype-read procedure.

If the English interpreter detects that the student's input is a question, or a question-like command, it will call the question-answering (Q/A) module, passing to it the input string together with an extra argument with value 1 which indicates that Q/A should process only that question, and then return. Upon return, the student is prompted to give the answer to SCHOLAR's former question, and we are back in the previous track. If at any time the English interpreter recognizes the student input to be the name of a general interaction mode (Q/A, etc.), it asks the student if he wants to change the mode of the interaction. Upon an affirmative answer, SCHOLAR changes the mode of the interaction to the requested one, and proceeds in this new mode. The student can also ask to terminate the interaction by typing EXIT or WRAPUP. Though not fully implemented in the current version, the WRAPUP type of exit is designed to present the student with a list of topics in which he needs further work.

The student also has the option to type at any time, even when it is not his turn to type, the symbol "#". SCHOLAR periodically checks the input buffer, and if it contains that symbol, an interrupt occurs, similar to that described above when the student typed a general interaction mode.

On the right-hand side of Fig. A.1 the large box represents the procedures for answering questions. After reading and interpreting the input, retrieval and/or other information-processing routines produce an answer, which is then converted into English sentences by the English-text generator. Then, the program usually loops to accept a new question. This is what happens in the Q/A mode of interaction. If in mixed-initiative, the question-answering routines return after a single question. Finally, the question-answering module can be accessed directly, and can also be exited either towards one of the other modes or towards termination.

Figure A.2 presents the principal components of SCHOLAR, both in terms of procedures and data types. The procedures are coded in terms of LISP functions. In Fig. A.2 we have classified SCHOLAR procedures in eight groups. These groups are directly related to the different functional modules which are the object of our analysis in Subsections A.2.d and A.3 to A.9 below. There is no one-to-one correspondence, however, since groupings used during programming are not optimized from a didactical point of view when a description of a complete system is desired.

FUNCTION GROUPS

- Interaction
- Context and Question Generation
- Student Input
- Sentence Generation
- Retrieval and Info. Processing
- Read and Print
- Auxiliary to Semantic Network
- Miscellaneous

DATA TYPES

- Fixed
  - Semantic Network
    - Examples
    - Application
    - General
  - Agenda
  - Messages
  - Miscellaneous
    - Operational
    - Initialization
- Dynamic
  - Context Push-Down List
  - Used-Question List
  - Certain Tags
  - Results
  - SetInteraction

Fig. A.2 Principal Components in SCHOLAR

The data types will be studied in some detail in Section A.2 that follows.

A.2 The Data Base

In this Subsection we will discuss the data base in some detail. In terms of the content of it, we have used Veliz (18), Aguilar (19) and Finch et al. (20) as our main references. In terms of the structure of SCHOLAR's data base, it centers around SCHOLAR's semantic network, and it seems natural to start with the detailed characterization of that network.

A.2a The Semantic Network

In previous parts of this document we have defined what a semantic network is, and given its principal characteristics. We have also discussed above some characteristics of the SCHOLAR's network (e.g., its organization in three levels). We specifically want to refer the reader to Section III.2. There, we defined some important concepts, namely those of unit of information, node, property, attribute, and value. We have also discussed already some important special attributes, like superconcept, and superpart.

We want to refer now to other details in SCHOLAR's network. Figure A.3 presents the concept unit latitude and fragments of the example unit Argentina. Let us observe that the overall organization of a unit is the same as that of a property. The first element identifies it, the second place is reserved for tags (and is NIL if they are absent). The rest (CDDR in LISP notation) is the value, and may contain atoms, atomic lists, procedures, and subproperties. This similarity between units and properties is not accidental, since it may be considered that a property is a unit which, instead of having a word as a name, has as such the semantic string formed by concatenating its attribute with the attributes of all properties in which it is embedded, till reaching the name of the unit to which it belongs.

Operationally, that similarity is convenient, because it simplifies the programming of routines that must deal with both units and properties. The similarity breaks down, however, to a degree, when we observe that the first element of a unit is more complex than just an attribute. Since a unit usually corresponds to a word, we must find a place to store what syntactic kind of word it is (i.e. what part-of-speech or POS it is), synonyms, semantic and syntactic markers, etc.

In order to do this, the first element (CAR in LISP notation) of a unit is formed by two lists. The first list contains the

```
(RPAQQ LATITUDE (((CN LATITUDE)
      (DET THE DEF 2))
    NIL
    (SUPERC NIL (DISTANCE NIL ANGULAR (FROM NIL
            EQUATOR)))
    (SUPERP (I 2)
      LOCATION)
    (VALUE (I 2)
      (RANGE NIL -90 90))
    (UNIT (I 2)
      DEGREES)))
```

```
(RPAQQ ARGENTINA (((XN ARGENTINA)
      (DET NIL DEF 2))
    NIL
    (SUPERC NIL COUNTRY)
    (SUPERP (I 6)
      SOUTH\AMERICA)
    (AREA (I 2)
      (APPROX NIL \ 1200000))
    (LOCATION NIL SOUTH\AMERICA (LATITUDE (I 2)
        (RANGE NIL -22 -55))
      (LONGITUDE (I 4)
        (RANGE NIL -57 -71))
      (BORDERING\COUNTRIES (I 1)
        (NORTHERN (I 1)
          BOLIVIA PARAGUAY)
        (EASTERN (I 1)
          (($L BRAZIL URUGUA
            NIL
            (BOUNDARY NIL URUGUAY\RIVER)))
```

```
    (CAPITAL (I 1)
      BUENOS\AIRES)
    (CITIES (I 3)
      (PRINCIPAL NIL ($L BUENOS\AIRES CORDOBA ROSARIO
            MENDOZA LA\PLATA TUCUMAN)))
    (TOPOGRAPHY (I 1)
      VARIED
      (MOUNTAIN\CHAINS NIL (PRINCIPAL NIL ANDES
            (LOCATION NIL (BOUNDARY NIL (WITH NIL
                  CHILE)))
          (ALTITUDE NIL (HIGHEST NIL ACONCAGUA
                (APPROX NIL 22000))))
        (SIERRAS NIL (LOCATION NIL ($L CORDOBA
                BUENOS\AIRES))))
      (PLAINS NIL (FERTILE NIL USUALLY)
        (($L EASTERN CENTRAL)
          NIL PAMPA)
        (NORTHERN NIL CHACO)))
```

Fig. A.3 The Units for Latitude and Argentina (Fragments) in SCHOLAR

POS followed by the name of the unit and contextually acceptable synonyms (e.g. height, elevation, and altitude). The part-of-speech can be: example-noun XN, concept-noun CN, adjective ADJ, adverb ADV, preposition PRP, modifier MODIF, determiner DET, verb VRB, pronoun PRN, and auxiliary AUX. The distinction between concept-nouns and example-nouns has operationally proved to be a necessary and convenient one.

The second list in the CAR of a unit is optional; it can contain a list of some semantic and syntactic markers with their values. One of them is DET, indicating the need for a determiner: for example, we say <u>the</u> U.S. but not <u>the</u> Uruguay. Another can be a marker indicating plural or singular: "Buenos Aires" is singular but may morphologically look like a plural, while the opposite happens with the word "people". Another marker can be DEF, with a numerical value, which, when present, locally overrides a system parameter which specifies the semantic depth set as a threshold to extract definitions from units.

SCHOLAR accepts names formed by more than one word, like Buenos Aires, South America, or Rio de la Plata. These are internally converted into a single atom by means of replacing spaces with backslashes which are again eliminated on output.

Figure A.4 presents an approximate Backus Normal Form (BNF) description of the syntax of SCHOLAR's internal representation. There, "First" is the CAR of the unit, while Posname is the first of its two lists, composed by POS and Namelist. Observe that the case in which Namelist is a positive integer has been added in order to handle the rare case of units with no name. Then, the POS and the number identify them.

Without trying to be exhaustive, let us look at some other details. We first see that there are some special names for important properties which appear with great frequency. None of them, however, is privileged in any sense, and all of them are optional. This is an important difference with Quillian's approach since he reserves the first place of a unit for the superconcept (which he calls superset) which is obligatory. This seems inconvenient because some words (e.g. many adjectives and verbs do not have a clearly defined superconcept).

Another point to note by the reader when examining Fig. A.4 is our definition of Atom', which can be either "any English word" (i.e., an atom), or several kinds of lists with atomic value. This allows the manipulation of those lists by the executive program as if they were atoms, until the time to either decompose or list them

```
<Unit>          →  <First><Taglist><Proplist> | NIL
<First>         →  (<Posname><Markerlist>)
<Posname>       →  (<POS><Namelist>)
<POS>           →  XN,CN,ADJ,ADV,MODIF,VRB,PRN,PRP,DET,AUX
<Namelist>      →  <Name> | <Name><Namelist> | <Positive Integer>
<Name>          →  <AEW>
<Markerlist>    →  (<Marker><Markervalue> <Markerlist> ) | NIL
<Taglist>       →  (<Tag><Tagvalue><Taglist> ) | NIL
<Marker>        →  "any one of various syntactic and semantic
                    markers"
<Markervalue>   →  <AEW><Markervalue> | NIL
<Tag>           →  I,GE,P,R
<Tagvalue>      →  <Number> | NIL    (Special restrictions depen-
                                      ding on Tag)
<Proplist>      →  <Prop><Proplist> | NIL
<Prop>          →  (<Propname><Taglist><Proplist>) | NIL |
                          <Atom'> | Function
<Propname>      →  <AEW> | <Sp-propname> | <Prop>
<Sp-propname>   →  Superc, Supers, Examples, General/Characteris-
                   tics, Applied/to, Properties, ...
<Atom'>         →  <AEW> | ($L<AEW><AEWlist>) | ($Q<AEW><AEWlist>)|
                    Booleval
<AEW>           →  "Any English word" | "any number" | "any
                    special term"
<AEWlist>       →  <AEW><AEWlist> | NIL
<Function>      →  ($F<Fname><Arglis>)  |  ($F<Lambda Expr>)
<Arglis>        →  NIL  | <Arg><Arglis>
<Booleval>      →  ($AND<Proplist>) | ($OR<Proplist>) |
<Pairlist>      →  (<Prop><Proplist>)<Pairlist> | NIL
```

Fig. A.4 Approximate BNF Representation of the Syntax of SCHOLAR's Semantic Network arrives. Incidentally, the list with $Q is a quoted list, while $L indicates a list of words syntactically equivalent, like a list of countries or rivers. Actually, the $Q-kind of Atom' is not currently used; it was introduced as a way of inserting pieces of text in the data structure if that was necessary. The capabilities of our English-text generator have made that unnecessary.

A final point we would like the reader to notice is the freedom with which the value of a property can be written. It is essentially a list of properties (proplist), which may be NIL. In that case, the semantic interpretation is that the attribute is true. If not NIL, the value may be a list of any number of atoms, or atomic lists (atom'), or subproperties. Atoms and atomic lists can obviously be considered as part of the value of the property, and also as terminal single elements of semantic strings which are true. For example, referring to Fig. IV.4, it is true that the topography in Argentina is varied.

An important item in SCHOLAR's operation on its semantic network is the <u>use of tags.</u> The program utilizes both permanent and temporary tags. Permanent tags are markers on items in the data base which we want to associate with the way knowledge is originally coded, rather than with a time-dependent utilization. Both kinds of tags have very different implementations; temporary tags do not appear as part of the data base, and will be considered in Subsection A.2c below.

The second place in each unit or property is currently reserved for a possible list of tags, and if none appears, NIL is inserted. A possibly convenient alternative is to consider tags as any other property of a more informational character, and, as those, they would be optional. Anyway, this is a possible subject for future explorations, rather than speculation now.

Though at some phase during the development of SCHOLAR we have included some permanent tags like P (for probability of occurrence) and R (for reliability of information), the only permanent tag being used by the current version of SCHOLAR is the irrelevancy tag I*. In each unit or property this tag is optional; if not present, it is given the value zero. It can have any of the seven integer values between 0 and 6, following Miller (21) and Quillian (2), and also because neither finer nor coarser resolution seemed to be preferable to the 7-point scale.

---

*A tag to signal for past tense is also occasionally used.

The irrelevancy tag I is used when determining the semantic depth which characterizes the relevancy of a node with respect to another node. For example, we can talk of the relevancy of property with attribute "plains" with respect to "topography of Argentina," or with respect to "Argentina" itself.

Figure A.5 is a partial diagram of the semantic network as seen from the node "Argentina". We see portions of the tree which is the unit "Argentina", other units like "country", "Bolivia", "country" again, etc. As a matter of fact, through the property "examples" of the unit "country" we could circle back to "Argentina" (and similarly through other paths).

In Fig. A.5 the abscissae represent semantic depths, measured as the sum of the number of embeddings, plus the sum of the I's in the traversed links. Since semantic depth is what we are using as a measure of semantic irrelevancy (SI), the horizontal axis is a measure of irrelevancy (more precisely, irrelevancy through a given path). Therefore, if we now want to extract the most relevant pieces of information, we can draw a verticle line at a give SI (say 2 or 3) and retrieve all paths that lead to terminal nodes located to the left of that line, i.e., with semantic irrelevancy less than the given threshold. If we want successively more and more information about "Argentina", we can retrieve successive bands of nodes, at increasing semantic depths.

Some specific examples of the use of the semantic depth will be presented in Sections A.5 and A.6 below. It must also be said that the tag I is also used to compute a weight for weighted random selection of questions (see Subsections A.2d and A.8 below).

A.2b Other Permanent Information

The data base in SCHOLAR contains other permanent entries not included in the semantic network. We must first mention lists of standard messages which SCHOLAR presents to the student under appropriate circumstances. There is a fairly extensive repertoire of about one hundred messages of all sorts, most consisting of one sentence, but some longer. For many of the messages, several alternatives are available; they are selected at random, with the provision that no single alternative can be presented two consecutive times.

Some messages allow a certain degree of construction, with portions that are filled, for example, with errors detected in the student's answers. A case in point is shown in the protocol of Fig. 1, part (e) where the computer responds to a partially wrong answer with

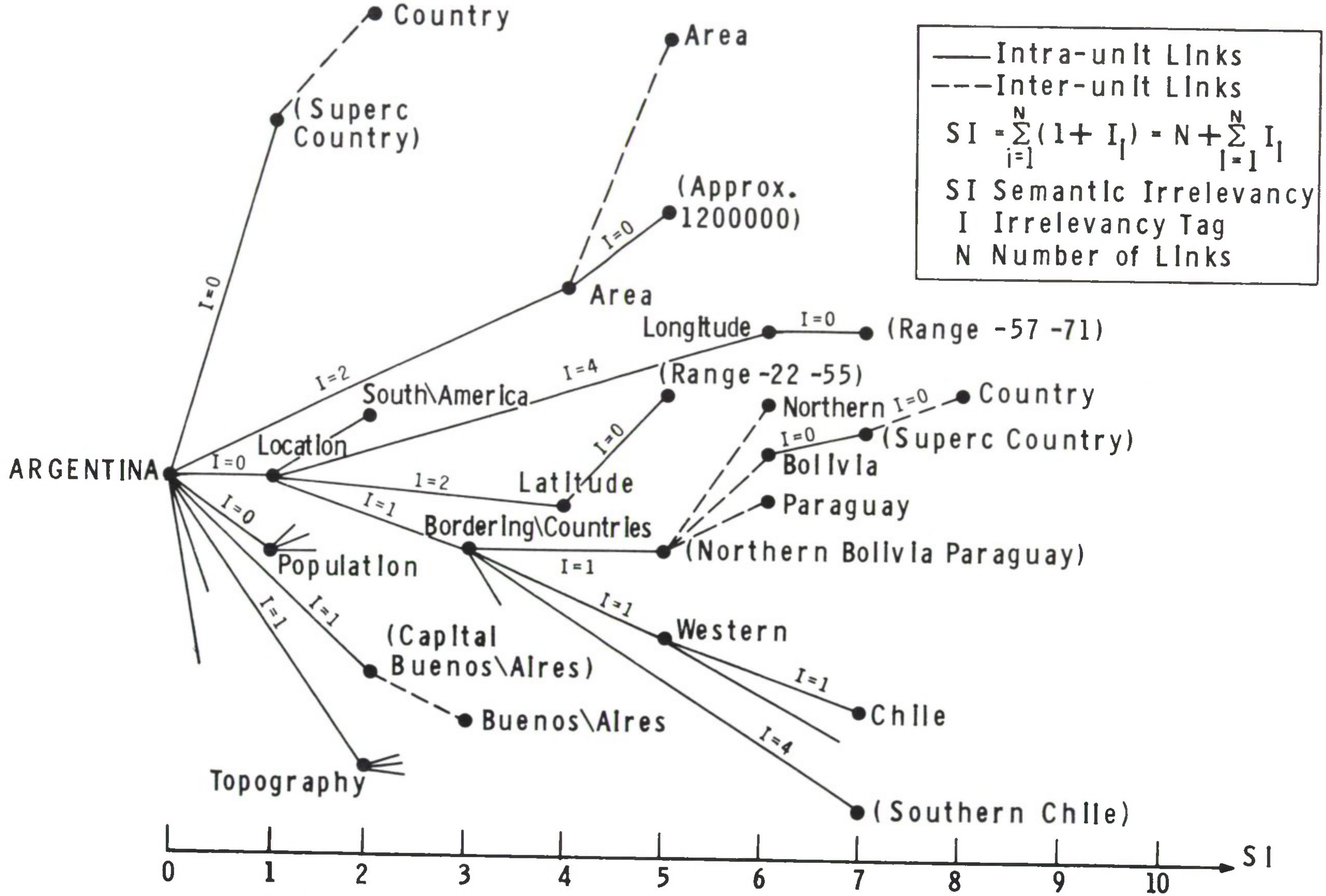

Fig. A.5 Semantic Irrelevancy for The Node "ARGENTINA" (See Text) the following semi-constructed messages:

> Haven't you incorrectly used "Bogota" and "Aconcagua"?
> On the other hand "La Paz' and "Sucre" is correct.
>
> The complete answer also includes Potosi, Cochabamba, and Santa Cruz.

Another important piece of information which is not modified during the student's interaction is the agenda of topics to be discussed during that interaction. This agenda is a plan that the teacher can specify with greater or lesser detail. In the most interesting case, only an overall context is given (South America in our case); all the rest will be dynamically generated by the computer. This was the case when all protocols and printouts forming part of this work were produced.

A heterogenous group of constants and lists used by different portions of the program must also be mentioned here. We have, for example, lists of interrogative words (like "?" and "tell\me), punctuation marks, synonyms, compound words, etc.

A.2c <u>Temporary Information</u>

A system like SCHOLAR uses temporary information to a considerable extent. That information is dynamically changed by the program. There are several major kinds of temporary information.

An important kind is represented by <u>temporary tags</u>. These tags refer to the information structure but <u>must not be a</u> part of it. Their examination and modification should also operate fast since such operations are done very frequently. Because of this, temporary tags are dealt with by means of <u>hash-coding routines</u> available in BBN-LISP. This hash-coding operates <u>on the virtual address</u> of a given item and translates it into an entry in an array previously declared. If the arrays are sufficiently large in order to be sparsely filled, the hash-coding routines operate fast and unobtrusively.

Four different temporary tags are manipulated this way. The first one, called CTXGEN/HSH, refers to the array of the same name and deals with the generation of contexts. Tagging avoids repetition of already used contexts if generated in a random way, but does not block them if triggered by some diagnostic or other error-related operation.

The second temporary tag is the most important, and is called A&E/HSH where A&E stands for "activation and error." This tag is applied to properties which have been used in questions by SCHOLAR.

When the question is first asked, a value 1 is assigned to the tag meaning that the question has been used. When an error is being investigated, then the tag becomes -1. Finally, if all subproperties of a given property have been used, so no more use can be mode of that property for question generation, it is tagged with a 0.

The tag called #QUES/HSH is inserted at the same time as the A&E/HSH tag, but its value is a number which identifies the question where that item was used. This permits to identify, from the data structure, past questions referring to certain portions of that data structure.

Finally, there is a temporary tag called HI'/HSH which is inserted at the time of a question by the student. It has as value the semantic depth used so far in retrieving information at that point, and permits the asking of questions of the "tell-me-more" kind.

All temporary tags apply to either units, properties, or atoms (including atomic lists) considered as true properties. Hash-coding the address of a property, or worse, of an atomic value, would affect all possible occurrences of that item, both in the correct context and otherwise. To avoid this undesirable effect, tagging is always done not on the item, but on the list which has that item as its first element. In LISP notation, instead of tagging an element of a list, we tag the CDR if the corresponding CADR is the element in question.

A very important piece of temporary information is the context push-down list (CTXPDL), which permits to keep track of active contexts. At the beginning of the program, the CTXPDL is set to the agenda, which we discussed above.

During execution of the program, the CTXPDL changes by modification or deletion of old elements, or addition of new ones. Each element includes the name of a unit or property (properly individualized) which acts as a context, plus other information. At any point of time the valid context is that corresponding to the element on top of the list. The bottom element contains a context which is considered as the overall context (or CTXØ). In all our experiments so far we have utilized a one-element agenda, with only an overall context in it; in other words CTXØ has consistently been South America. A system parameter (set in advance in SETINTERATION by the teacher, see A2.2 below) is the approximate overall duration of the session. This is a number DURØ in the agenda, which yields another number in the CTXPDL which relates to the approximate time of day, in seconds, at which the session must end.

When a new context is added in front of the CTXPDL, a life which depends on its relevancy is assigned to it and a time at which its use should terminate is set. At certain points in the program (for example, before generating a question) the CTXPDL is examined by a procedure called PERCHK and pruned of all contexts whose life has expired.

Figure A.6 shows the state of the CTXPDL at some stage in the program, together with QUESLIS, the question-used list to be discussed below. We see that the CTXPDL has two elements, respectively headed by "Guyana" and "South America." In the second we see in the list following the word CTXGEN that three contexts (Paraguay, Colombia, and Guyana) have been generated, the numbers being their semantic depths with respect to South America. Of these three contexts, Paraguay and Colombia have already been erased (possibly with some subcontexts), and only Guyana is alive. The questions asked about Guyana are detailed in the element headed by that name; for each of them we successively have the number of the question, the expected answer, the semantic string on which the question is based, and the mode of the question. Incidentally those two questions are 11 and 12 by the system, but 14 and 15 when we include student questions (see variable #QUESINTERRL in CTXPDL).

In the same Fig. A.6 we see a fragment of QUESLIS, the list of questions already asked by either SCHOLAR or the student. Actually the three questions referred to in Fig. A.7 were asked by SCHOLAR whose name appears there; otherwise we would have had the word STUQ.

QUESLIS is not a push-down list, so information is not erased from it. It is added, however, in two steps: the first when the question is formed, the second when the answer is evaluated. Question 15, the last one, is in the phase between formulation and answer evaluation, since evaluation of the student's answer has not yet been inserted; in other words, the program was interrupted after formulating that question, and the printout of CTXPDL and QUESLIS obtained. The information inserted after evaluation of the answer incidentally, the value returned by MATCH1 (see subsection A.9 below).

The second number, after the word SCHOLAR, represents the questions asked by it; the difference with the first number corresponds to the number of questions asked by the student.

Observe that each question keeps tract of its context, and after QUESINTERRL the numbers of questions within that context can be found.

```
(RPAQQ CTXPDL ((GUYANA (LIF 4673 DI 3 DURØ 12ØØ
          QUESINTERRL (14 15))
       (11 (T TRUE YES Y OK CORRECT RIGHT
            CORRECT\TO\SAY RIGHT\TO\SAY OK\TO\SAY)
         (SOUTH\AMERICA LOCATION GUYANA)
         T/F NIL)
       (12 (COUNTRY)
         (COUNTRY SUPERC GUYANA)
         WH NIL))
     (SOUTH\AMERICA (DURØ 36ØØ DI Ø LIF 4Ø53
          CTXGEN (PARAGUAY 3 COLOMBIA 3 GUYANA 3)))))



(RPAQQ QUESLIS (((1 SCHOLAR 1)
       (CTX PARAGUAY QUESINTERRL (1 2 3 4)
         #QUES 1 DI 5)
       (((SOUTH\AMERICA LOCATION PARAGUAY)
           NIL)
         PARAGUAY FILL-IN)
       (WRONG (SOUTH\AMERICA)
         (VENEZUELA)
         NIL
         (SOUTH\AMERICA)
         (VENEZUELA)
         (VENEZUELA)
         NIL
         (MISP APPROX)))
```

```
     ((14 SCHOLAR 11)
       (CTX GUYANA QUESINTERRL (14 15)
         #QUES 11 DI 5)
       (((SOUTH\AMERICA LOCATION GUYANA)
           NIL)
         GUYANA T/F)
       (CORRECT (T TRUE YES Y OK CORRECT RIGHT
            CORRECT\TO\SAY RIGHT\TO\SAY OK\TO\SAY)
         (YES)
         (YES)
         (T TRUE Y OK CORRECT RIGHT CORRECT\TO\SAY
            RIGHT\TO\SAY OK\TO\SAY)
         NIL NIL NIL
         (LIST MISP APPROX)))
     ((15 SCHOLAR 12)
       (CTX GUYANA QUESINTERRL (14 15)
         #QUES 12 DI 5)
       (((COUNTRY SUPERC GUYANA)
           NIL)
         GUYANA WH)
       ((COUNTRY SUPERC GUYANA)
         WH))))
```

Fig. A.6 The Context-Push-Down List and the Used-Question List

The QUESLIS can be accessed from the data base through the temporary (hashed) tag #QUES/HSH discussed above. Thus, from any property, we can find a number which indicates what question has dealt with it.

A.2d Some Auxiliary Procedures

It is pertinent to mention here some auxiliary routines that operate in direct relation with the data base, and are dependent on its configuration. Some of them are the functions that check on the CTXPDL, namely PDLCHK and PERCHK which calls the former. Actually, PERCHK also performs other checks, like looking for an interrupt call by the student when it is not his turn to type, and for the imminence of a LISP garbage collection. In the latter case, it would be bad for that system operation to occur in a inappropriate place (like the middle of a type-out) since the garbage collection can take up to a few minutes for a system of the SCHOLAR's size. Also the garbage collector prints a message related to words collected and available, completely meaningless and distractive for the student. For that reason, a procedure GCMESS is called if a garbage collection is bound to occur soon, forces it to occur immediately, and blocks the system message replacing it by another telling the student to wait for a while (see Fig. 1(c): "Wait a minute...").

Other important auxiliary procedures related to the data base are those for manipulating tags. Permanent tags are read by TAGCHK, a LISP function with two arguments which are the property, and the name of the tag; TAGWRT can write permanent tags, as a function of three arguments: property, tag name, and value.

Temporary tags are manipulated by similar functions, now called TAGCHK' and TAGWRT', which use the hash-coding routines, and where the name of the tag points to the corresponding storage array. As we said above, they act on the list which has the property as its first element rather than on the property itself. A third function, TAGWRTØ, can write a tag as TAGWRT', with the additional feature that it checks also other properties at the same level of the one just tagged; if all have already been tagged, it tags the upper level, where the procedure is recursively repeated. This avoids wasting time in future searches.

For the task of performing weighted random selection of a string in the data base, the basic function is called SELECT. It examines all the items that form the value of a property or the informational part of a unit (i.e., their CDDR in LISP notation). SELECT retrieves for each one a weight which is the difference between 6 and the value of the irrelevancy tag I for that item; if the item is an atom, an atomic list (see above), or if the tag is NIL, SELECT considers the tag as Ø, and the weight as 6. The selection of an item is then done probabilistically using the weights thus obtained. There are also mechanisms for optionally disabling the weights and considering all items as equally relevant.

We have referred above to a number of system parameters which regulate the operation of SCHOLAR in its interaction with the student. Some of the most important system parameters (in a very general sense of the word parameters, some may be complex symbolic lists) can be set by the teacher by using a special interactive program called SETINTERACTION. Figure A.7(a,b) is an on-line protocol taken during utilization of SETINTERACTION. We see that the teacher does not have to know any LISP, or use any cryptic computer language. The program makes suggestions in English and guides the teacher in each step. The only requirement is for him to have some very general understanding of SCHOLAR and its parameters.

With respect to the set of parameters dealt with in Fig. A.7, it must be said that they do not represent, by far, an exhaustive list of the adjustable system parameters in SCHOLAR. For example, the specific tolerances for accepting an approximate numerical answer or a misspelled word depend on adjustable parameters (see Subsection A.9 below). The question generation routines select question modes according to pre-established weights which can be adjusted. Or we may think of even more detailed parameters, like that regulating the probability of generating unrelated alternatives in multiple-choice questions. Though we could add these to the list of parameters set by SETINTERACTION, it may be too much of a burden and too difficult a task for a teacher to have so many degrees of freedom. We do not know the optimal answer to this question which may have interesting pedagogical implications. One possible solution may be to have two levels for SETINTERACTION, one that is easily handled by the fairly naive instructor, and another which may refer to more fundamental and/or detailed questions and which may require greater expertise.

A.3 Read, Print, and Other Interactive Procedures

In many CAI and other interactive systems, there are systems-imposed limitations in input-output. Typical, for example, are for the user to have to limit his input to one line, to have to read computer output with unnatural places for punctuation marks (like always separated by a space from the previous word, or appearing at the beginning of a line of output), to have "yes." accepted as a correct answer but "yes" rejected as such, or to be artificially forced to form single words for terms like South America or Rio de la Plata. Though these are not conceptually important problems, they do impair through

```
←SETINTERACTION)

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

 THIS IS THE PROGRAM TO SET THE CONDITIONS OF THE INTERACTION
       BETWEEN THE STUDENT AND SCHOLAR. DO YOU YOU WANT
       TO CHANGE THOSE CONDITIONS? PLEASE TYPE Y OR
       N. (REMEMBER TO TERMINATE YOUR TYPING WITH AN
       ASTERISK * FOLLOWED BY A CARRIAGE RETURN.)

*Y*

 NAME OF INSTITUTION:

*ABC REGIONAL HIGH SCHOOL*

 TYPE NAME OF SUBJECT MATTER, I. E., CONTEXT TO BE DISCUSSED:

*GEOGRAPHY OF SOUTH AMERICA*

 TYPE OF INTERACTION. IT MUST BE ONE OF THE FOLLOWING:
       MIXINIT, TEST, OR Q/A.

*MIXINIT*

 INSTRUCTOR IN CHARGE OF THE COURSE:

*MR. JUAN ECHEVERRIGARAY*

 TYPE YOUR NAME EVEN IF YOU HAVE TYPED IT ABOVE:

*JAIME CARBONELL*

 TODAY'S DATE:

*4/15/1970*

 MAX. DURATION OF STUDENT INTERACTION, IN MINUTES:

*60*

 MIN. DURATION OF STUDENT INTERACTION, IN MINUTES:

*40*

 MIN. NO. OF QUESTIONS TO BE PRESENTED:

*15*

 IF YOU WANT FULL INITIAL INSTRUCTIONS PRESENTED AT
       THE BEGINNING OF THE INTERACTION WITH THE STUDENT,
       TYPE 1. IF NOT, TYPE 0.

*0*

 LET US NOW DECIDE IF SCHOLAR SHOULD CALL THE STUDENT'S
       ATTENTION ABOUT WORDS IT CAN NOT RECOGNIZE. PLEASE
       TYPE Y OR N:
```

Fig. A.7 (a) On-Line Protocol of Teacher Using SETINTERACTION in SCHOLAR (CONT)

```
*Y*

 TYPE PROBABILITY, IN PERCENT, FOR GENERATING A QUESTION
       ABOUT A SUBCONTEXT OF A GIVEN CONTEXT, WHEN DEALING
       WITH THE CONTEXT ITSELF:

*25*

 TYPE THE NUMBER OF SECONDS TO WAIT BEFORE PRODUCING
       A PROMPTING MESSAGE:

*20*

 TYPE MAXIMUM SEMANTIC DEPTH ACCEPTABLE FOR SUBCONTEXT
       GENERATION:

*6*

 SCHOLAR IS SET BOTH TO CHECK FOR MISPELLINGS IN THE
       STUDENT'S ANSWERS AND TO ACCEPT APPROXIMATE NUMERICAL
       ANSWERS. NORMALLY YOU WILL WANT TO LEAVE BOTH
       OF THESE CHECKS IN. YOU DO THIS BY TYPING ---
       WITHOUT THE QUOTATIONS, OF COURSE ---: "MISP
       APPROX". IF YOU ONLY WANT ONE OF THEM, TYPE ITS
       NAME. IF YOU DESIRE NONE, TYPE NIL.

*MISP APPROX*


 DO YOU WANT TO START THE STUDENT INTERACTION NOW? ANSWER
       Y OR N.

*N*

 O. K. THE VALUES YOU HAVE ENTERED HAVE BEEN STORED
       IN THE SYMBOLIC FILE /SETINTER/.


++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Fig. A.7 (b) On-Line Protocol of Teacher Using SETINTERACTION in SCHOLAR

(CONCLUDED)

extra constraints the tendency towards free and comfortable interaction.

BBN-LISP read/print facilities were inadequate for our purpose, so a new read/print package was implemented. Input and output can be text of any length. Names previously declared as composed of two or more words are automatically transformed on input into a single atom by replacing blanks with backslashes. Internally they always maintain those backslashes, but on output, backslashes are replaced back by blanks.

Punctuation signs are separated from the preceding word or element (and from what follows). This is needed in order for the words themselves to act upon the semantic network. On the other hand, we do not accept the obvious solution of filtering out the punctuation marks in the reading program. The question mark is one of the possible interrogative words indicating a question; other punctuation marks may be important in possible language applications, and even for language comprehension in future versions of SCHOLAR. The reading routines also detect comparatives and superlatives on input and transform them appropriately (though not all the procedures to deal with comparatives and superlatives are operational at this time).

Many auxiliary routines associated with either printing or reading have been coded. One of the auxiliary routines associated with printing is PRAND, which, given a list (X Y Z) of items, prints it out as: "X", "Y", and "Z". PRCOL prints a list as a column of items, and is used, for example, in multiple-choice questions.

An important routine associated with reading by SCHOLAR is called PABLO; it handles the changes in control from SCHOLAR to the student and vice-versa; it operates by calling the basic reading routine RD*. While waiting for input, PABLO measures elapsed time; if this exceeds a given delay, PABLO prompts the student to respond, then records the excessive delay. PABLO has a delay threshold which applies to delays measured before the student begins his typing, and a longer one for the total time before return of control. If the latter is exceeded it complains about the delay, and again records it. Finally, while PABLO is waiting for input, and in order to avoid excessive central-processor utilization, the whole program is dismissed for fixed amounts of time, now set at 1 sec. This is an interval which seemed reasonable in terms of man-computer interaction.

A.4 The Retrieval Procedures

A fundamental component in an ISO CAI system is the group of procedures for selectively retrieving information from the data base (the semantic network).

Generally, retrieval procedures in SCHOLAR are handled by means of the use of an intermediate language consisting of attribute-object-value triples. These three elements are the first three arguments of the top retrieval procedure called RET, which has a list of flags as an optional fourth argument.

Figure A.8 shows in LISP EVAL notation, the different cases which we may have (the fourth argument has been omitted for simplicity). After the first general line, we have the most usual case, when the value is sought. This internally translates into a call to the procedure TETV (for "retrieve value"). The second case occurs when the object is sought, with a call to the procedure RETO (for "retrieve object"). In the third case the attribute is the unknown; this internally translates into a call to the procedure RETA (for "retrieve attribute").

In the fourth case (fifth line) all three arguments are given corresponding to a true-false question. SCHOLAR uses object and attribute to retrieve a value from the semantic network. The comparison of this retrieval value with the given one answers the true-false inquiry.

In the fifth case (sixth line), both attribute and value are unknown, as in the question "Tell me about Peru." The sixth and seventh cases are special ones of a rather pathological nature (they respectively correspond to commands to retrieve all instances in which a given Z appears as a value, or all instances in which a given X appears as an attribute); they require extensive searches, and need not be of further concern to us.

The classification above tacitly assumes that elements in the triple are well-defined atomic values. This is not always the case, but a simple generalization provides the solution. The attribute, for example, can frequently be the concatenation of several atomic values, as in the questions: "What is the form of government in Uruguay?," and "Give me the principal countries of origin of the population in Argentina." In these cases, the attributes are respectively extracted as (form government), and (principal countries origin population). Processing these cases is done by an intermediate procedure RETØ, which recursively calls itself with an atrribute obtained by removing the last element of

```
(RET   ATT   OBJ   VAL)
       (RET   ATT     OBJ     QMARK)
       (RET   ATT     QMARK   VAL)
       (RET   QMARK   OBJ     VAL)
       (RET   ATT     OBJ     VAL) ——→ T/F Case

       (RET   QMARK   OBJ     QMARK)

       (RET   QMARK   QMARK   VAL)  }  Special
       (RET   ATT     QMARK   QMARK) }  Unusual
                                        Cases
```

NOTES: 1) ATT, OBJ, VAL can be recursive calls to RET
2) ATT, OBJ, VAL can be conjunctive sets.

Fig. A.8 Triples in the Retrieval Package of SCHOLAR the original attribute, and an object which is the result of applying RETØ with the last element of the original attribute as attribute, and the original object as object.

Incidentally, the discussion above shows that the object may not only be the name of a unit, but also the tree-list which is the unit itself, or any of its properties or subproperties; these different cases are automatically handled by the retrieval package of routines in SCHOLAR.

Let us now consider the case in which the object is not a tree-list but a list of depth 1 obtained by the concatenation of attributes and an object. In effect, we can generalize here our notion of names. Any string that points unambiguously (in the sense of retrieval capability) to a unit or property (i.e., a node in the semantic network) can be considered as a name for that unit or property. Thus if the object is a list of attributes and an object, RETØ with the first element of the object as attribute (or that element appended to an existing attribute, if not null) and the rest (CDR) of the object as new object.

The fundamental internal procedures in the retrieval package are called RETX1 and RET-1. The former is a LISP function which takes an object either by name or as the tree-list itself, a maximum semantic depth, and a minimum one, and returns a tree-list of all the properties and subproperties that have irrelevancy in the prescribed semantic-depth range. In order to retrieve all available information it is enough to set the minimum depth to zero, and the maximum depth to a fairly large value, say 100. If we want only some information about a given object, or some definition of it, then in SCHOLAR the maximum is set to 2. This is the case in questions like "Tell me about Montevideo," or "What is Montevideo." Before returning its value, RETX1 writes a temporary tag which indicates the semantic depth at which further retrieval should proceed when and if requested. That would be the case with a question like "Tell me more about Montevideo" following one of the previous ones. In this case, a new layer of information, again 2-unit deep, would be retrieved, and so on. If at any time we ask "Tell me all ..." then all remaining information would be provided. To facilitate handling these various situations, a number of auxiliary functions like RETDEF, RETMOR, and RETALL exist; they do what their names suggest.

Figure A.9 shows the effect of RETX1 on a simple concept unit, that for "height." In (a) the internal representation is shown. In (b) we present two successive layers of output related to the unit "height." Instead of giving the tree-list representation, we give the English output as it would be presented to a student (see Subsection A.6 below).

```
(HEIGHT
    (((CN ALTITUDE HEIGHT ELEVATION) (DET THE))
    NIL
    (SUPERC NIL (DISTANCE NIL VERTICAL (ABOVE NIL SEA))
    (SUPERP (I 6) TOPOGRAPHY)
    (APPLIED\TO (I 2) MOUNTAIN CITY REGION COUNTRY)
    (UNIT (I 1) FEET)
    (VALUE (I 2) (RANGE NIL -1000 30000))))
```

Fig. A.9 (a) Structure of the Concept Unit "Height"
(CONT)

```
← RETDEF (HEIGHT)
  A HEIGHT IS A VERTICAL DISTANCE ABOVE THE SEA.
  THE UNITS ARE FEET.

← RETMOR (HEIGHT).
  HEIGHT IS APPLIED TO MOUNTAIN, CITY, REGION, AND COUNTRY.
  THE VALUE RANGES FROM -1000 TO 30000.
```

Fig. A.9 (b) Successive Layers of Output Related to The Unit "HEIGHT" (CONCLUDED)

RETX1 is called not only when retrieving information about an object, but also in the most frequent case of retrieving a value through RETV. If all existing information is desired in this case, an optional argument in RETV can block the call to RETX1.

The basic function utilized by both RETV and RETA is RET-1. It applies to an object which is a free variable for RET-1. Its only argument is an atom for which it searches that object. That search is performed by means of a call to the BBN-LISP editor which through a matching technique locates the atom. The return is a complex list which contains as first element the list in which that element is the CAR, and as successive elements the increasingly larger lists in which the first list is embedded, till the top level, i.e., the object, is reached.

In order to retrieve a value, RET calls RETV, which in turn gives the attribute to RET-1. RET-1 searches for this attribute which may be at the top level or at any depth within the object. RETV extracts the information it needs from the first element of the list returned by RET-1, and usually (unless this is specifically blocked) processes it by calling RETX1 before returning. Another function performed by RETV is that of handling plurals and singulars in the attribute, so, if the search by RET-1 for either form fails, an attempt with the other is made. This last feature permits more flexibility in both coding information and question making.

RETA collects the CAR's of all the elements of the output of RET-1, which are the different attributes leading from object to value. RETA is thus responsible for answering questions like "Montevideo is the --- of Uruguay," or "What is the relationship between the Aconcagua and Argentina?"

The function RETO is capable of finding the object of a triple in a question like "In what country the capital is Brazilia?" In this case, RETO is called, and searches to find which country in the list found in the property labelled "country" in the unit satisfies the question. RETO can also handle the more difficult question "Brazilia is the capital of ---." Here, the procedure must first start by finding what a "capital" is "applied/to" as an attribute in the unit "capital." The retrieved value is "country," and from here on we are back in the previous case.

Finally, true-false questions are processed by treating them as value questions (i.e., using RETV) and then comparing the proposed and the retrieved value by means of the same matching procedures used in evaluating student answers (see Subsection A.9 below).

The retrieval function now is RET-TF. Since the operation of RET-TF is closely tied to the form of the input, more on it will be said in the following subsections. (See A.5 and A.7.)

A.5 Processing Student Input

The student input can be an answer to a question by SCHOLAR, a question to SCHOLAR, or a command requesting either for a change in the overall mode of operation or for termination of the interaction. In the mixed-initiative mode any of the above forms of student input are possible when SCHOLAR passes control to the student.

Figure A.10 presents a particular example of some of the stages which are necessary to process student input. The first stage is really performed by the read routines. They take care of compounding words like "tell me" into tell\me, and also of separating punctuation marks from words. From this point on, the procedure called E-3 (for "English-to-triple") takes over. The first thing E-3 does is to check if the student input is the name of one of the modes of operation. If that is the case, it conducts an interchange with the student and sets the change in mode. If not, E-3 then calls CLEANQ (for "clean question") a procedure responsible for removing from the input courtesy words (like please and kindly), determiners and some other auxiliary words, and punctuation marks, except the question mark. Next, if the mode is Q/A, E-3 processes conjunctive elements (see Fig. A.11), but it must be said that the further handling of conjunctives by the present version of SCHOLAR is not yet completely operational. Next, E-3 looks for quantifiers (like one, three, more, something, everything\else), and puts them in a list of flags, together with some system flags like "misp" and "approx."

After that, E-3 searches in the transformed input in an attempt to find unbound words, i.e., words that have no meaning to SCHOLAR. This operation can be inhibited by a system parameter as is the case when an answer to a question by SCHOLAR is analyzed. If activated (as we have had it in our experiments), two lists are formed, one with bound and another with unbound words; the former is further purged of words that, though not defined in the semantic network, belong to a list REMQL formed by items like interrogative words, conjunctions, etc. If the reduced unbound list is not null, it is presented back to the student, and reformulation of the question is asked from him.

After all these stages, E-3 examines the pre-processed input to see if the statement being analyzed is a question, by checking

```
Please, tell  me something about the topographies of Argentina and Uruguay.
Please, tell\me something about the topographies of Argentina and Uruguay " . "
Tell\me something topographies Argentina and Uruguay
Tell\me something topography Argentina and Uruguay
Tell\me something topography ($and Argentina Uruguay)

    Tell me → Question, it originates a call to RET with
        Obj  =  ($and Argentina Uruguay)
        Att  = topography
        Val  = ?
        Figq = (something)
```

Fig. A.10 Processing Student Input

for the presence of one or more interrogative words or the presence of a "blank" word indicating a fill-in question (several standards for blanks are available). If not, the statement is considered as a response if in MIXINIT or TEST, and impossible to process if in Q/A. In this and other case in which the statement cannot be properly interpreted by SCHOLAR, it declares its incapability to understand the statement, and asks the student to rephrase it.

Usual questions (those using a question mark) and other interrogative statements (with tell me, etc.) are processed in similar ways, but processing of fill-in questions must follow a separate path in E-3 because of the different construction (which in fill-in questions is that of a complete affirmative sentence with one or more words replaced by a blank word).

Rather than a systematic parsing of the pre-processed input, E-3 uses a mixture of keywords and forms with detailed characterization of types of questions. In a sense, it searches a tree of characteristics which progressively narrow down the possible alternatives. At some point, E-3 passes tentative arguments to RET. In some cases, this is not a definite commitment, since if RET fails, the failure is communicated back to E-3, which, if possible, may attempt an alternate path. This is the case, for instance, in RET-TF with some alternative constructions for true-false questions, like "Is it true that Santiago is the capital of Chile?" In some cases E-3 can arrive at the same result while processing different equivalent questions like the different types of true-false questions: "Is Santiago the capital of Chile?," "IS it true that Santiago is the capital of Chile?," and "Tell me if Santiago is the capital of Chile." These would all originate identical results though their construction is quite different.

We do not think it would be interesting to specify here in detail the tree of tests used by E-3 in its attempts to "comprehend" a question. Let us just end this subsection by saying that if E-3 cannot obtail an answer for a question which it considers well-formed and which contains no undefined terms, it assumes that the requested information is not part of its semantic network. In that case, SCHOLAR confesses its ignorance by typing out a message like "Sorry, I don't know." This heuristic has proved satisfactory in most cases.

A.6 Generation of English Text

The procedures for the generation of text in English are an important module of the SCHOLAR system. In the situation of reviewing existing knowledge the need for text generation arises mainly in response to students' questions. In a teaching situation that need also exists when presenting new material.

Let us emphasize again that SCHOLAR never uses a transformation of the English question à la Eliza (Weizenbaum, 13) to produce its response. SCHOLAR completely constructs the sentences on the basis of what it extracts from memory (perhaps after due processing).

Our approach to the generation of English text has been to properly deal with all probable rather than all possible sentences. It soon became clear that we could avoid many complex characteristics of natural English by considering what characteristics the sentences to be generated would have. Further extensions and perhaps some revisions can always be made.

The English-text generator is based on the following constraints:

(1) No embedded clauses

(2) Generation of the simplest possible sentences. Whenever the material available is enough for a reasonable sentence, make it, and delete that material. This means that when the information is complex, several sentences are produced, in some cases with suitable pronominalization.

(3) Limited repertoire of verbs. The verb is generally "be," with some other possibilities (like range, average, etc.).

Constraint (2) leads to a left-to-right, single-pass generator. Constraint (3) was adopted after some study of possible cases. For example, the sentence

Uruguay produces principally wool and meat.

can be expressed as:

The principal productions of Uruguay are wool and meat.

Figure A.11 shows some examples of output, originated by questions of Uruguay. The reader will notice the acceptable quality of the English produced. Also observe the following characteristics: indentations which are related to the relevancy of different items, a couple of cases in which pronominalization has been applied,

```
←

←Q/A)

*TELL ME ALL ABOUT THE GOVERNMENT OF URUGUAY.*

   I THINK I HAVE ALREADY TOLD YOU ABOUT IT.

   THE GOVERNMENT IN URUGUAY IS STABLE, DEMOCRATIC,
       AND REPUBLICAN.
     IT IS NOT MILITARY.
     THE FORM IS UNITARY.
     THE PROBLEMS ARE ECONOMIC.
     UNREST IS SOCIAL AND STUDENT.
     THE REVOLUTIONS ARE NONE.


 *TELL ME ALL ABOUT THE LOCATION OF URUGUAY.*

    THE LOCATION IN URUGUAY IS SOUTH AMERICA.
        THE LATITUDE RANGES FROM -30 TO -35 DEGREES.
        THE LONGITUDE RANGES FROM -53 TO -58 DEGREES.
        THE NORTHERN AND EASTERN BORDERING COUNTRY IS
            BRAZIL.
        THE WESTERN BORDERING COUNTRY IS ARGENTINA.
          THE BOUNDARY IS THE URUGUAY RIVER.
        THE SOUTHEASTERN COAST IS THE ATLANTIC OCEAN.
        THE SOUTHERN COAST IS THE RIO DE LA PLATA.


*TELL ME ALL ABOUT THE POPULATION OF URUGUAY, PLEASE.*

   WE HAVE ALREADY DISCUSSED THIS.

     THE POPULATION IN URUGUAY IS APPROX 2800000 PEOPLE.
       ORIGIN OF THE POPULATION IN URUGUAY IS FROM EUROPE.
         THE PRINCIPAL COUNTRIES OF ORIGIN OF THE POPULATION
             IN URUGUAY ARE SPAIN AND ITALY.
     THE RACE IN URUGUAY IS WHITE.
       THE COMPOSITION IS WHITE 98 PERCENT.
       IT IS INDIAN 0 PERCENT.
     THE LITERACY OF THE POPULATION IN URUGUAY IS 95
         PERCENT.
     THE LANGUAGE IN URUGUAY IS SPANISH.
       THE RELIGION IN URUGUAY IS NOT OFFICIAL.
       THE PRINCIPAL RELIGION IN URUGUAY IS CATHOLICISM.
       SECONDARY OF THE RELIGIONS IN URUGUAY ARE JUDAISM,
           PROTESTANTISM, AND AGNOSTIC.


 PLEASE WAIT. I'LL BE BACK IN A MINUTE.

 OK. LET'S CONTINUE.

 *
```

Fig. A.11 English-Text Generator Operating on Answers About Uruguay correct subject-verb agreement, and especially the way information is broken down into sentences.

The English-text generator has performed very satisfactorily so far. We do not anticipate difficulties as long as the data base is properly constructed in relation to English input; this will be the case when an author language is developed. The English-text generator would only find difficulties if the data base is artificially created as a capricious set of synoptic trees representing the knowledge about the units; in this case, on the other hand, a human would encounter similar difficulties in generating English.

Figure A.12 presents the English output together with the internal representation corresponding to an answer to a question such as: "Tell me everything about the topography in Argentina." Observe here similar features as in Fig. A.11 as well as some new ones. For example, pronominalization appears again. Subject-verb agreement is apparent. Observe also the alternative use of "on" or "in" after "located." The right preposition is selected on the basis of a semantic marker in the head noun of the predicate, which depends on its shape. (A boundary is a line, but Cordoba and Buenos Aires are regions).

With respect to "plain" vs. "plains," only the latter appears in the internal representation. It is singularized into "plain" whenever required by the overall sentence. We will shortly discuss further the generation of the sentences related to the plains of Argentina (see Fig. A.13 below).

Finally, an interesting capability of SCHOLAR is the insertion of the unit "feet" after the number 22000. This unit does not explicitly appear in the internal representation. Having found a number, SCHOLAR searches for the closest concept-noun to which it might relate. If that noun (in our example it is "altitude") has a unit, it is extracted and added after the number.

Let us now discuss the procedures used to obtain the results shown above. The top procedure is INT-E (for "internal-to-English") which performs some initialization and checks, and prepares the call of INT-E-Ø, the real working horse. INT-E-Ø is responsible for breaking the tree-list of information taken from the semantic network down into smaller strings. INT-E-Ø accomplishes this by recursive calls to itself, till the strings are appropriate to produce English sentences. The supervision of the construction of individual sentences is done by INT-E-SENT (for "internal-to-English-sentence"), except when certain special attributes like location, superc, superp, range, and average, are found. These

```
 E (INT-E X Y 0)

   THE TOPOGRAPHY IN ARGENTINA IS VARIED.
     THE PRINCIPAL MOUNTAINS ARE THE ANDES.
       THE ANDES ARE LOCATED ON THE BOUNDARY WITH CHILE.
         THE HIGHEST ALTITUDE IS THE ACONCAGUA.
         IT IS APPROX 22000 FEET.
       THE SIERRAS ARE LOCATED IN CORDOBA AND BUENOS AIRES.
     THE PLAINS ARE USUALLY FERTILE.
     THE EASTERN AND CENTRAL PLAIN IS THE PAMPA.
     THE NORTHERN PLAIN IS THE CHACO.
NIL
←

 EDITV(X)
EDIT
*
 PP
  (VARIED (MOUNTAINS NIL (PRINCIPAL NIL (ANDES NIL (LOCATION NIL
              (BOUNDARY NIL (WITH NIL CHILE)))
          (ALTITUDE NIL (HIGHEST NIL ACONCAGUA (APPROX NIL 22000)))))
      (SIERRAS NIL (LOCATION NIL ($L CORDOBA BUENOS\AIRES))))
    (PLAINS NIL (FERTILE NIL USUALLY)
      (($L EASTERN CENTRAL)
        NIL PAMPA)
      (NORTHERN NIL CHACO)))
*OK
X
←E Y
(TOPOGRAPHY ARGENTINA)
←
```

Fig. A.12 Output of English-Text Generator and Internal Representation Related to a Complex Property attributes require and deserve special constructions. They originate a call to the procedure SPATT (for "special-attributes").

INT-E-SENT calls several other procedures. One of them is called ATTØ; it handles the relation of the present potential subject (it has been prepared by INT-E-Ø) with previous ones, and may decide to modify or pronominalize it. OF-IN-ON takes the string which is going to be the subject on the sentence, and forms a phrase with properly placed determiners (which are added), adjectives, and prepositions connecting nouns. It also handles atomic lists of nouns or adjectives to produce English conjunctive phrases. In all this, OF-IN-ON is the principal routine with help from many lower-level ones to perform the different specific tasks.

The procedure VRB (for "verb") is next called by INT-E-SENT; it selects the appropriate verbal form, which can be singular or plural depending on both subject and predicate; incidentally, VRB can, if necessary, modify the number of the tentative subject of the sentence in order to preserve agreement with the predicate. VRB can also use past forms if this is indicated to it by a flag.

Finally, the routine INT-E-PRED, through recursive calls to itself, and with the help of different lower-level routines, constructs the predicate for the sentence.

Many different auxiliary procedures had to be developed in support of English-text generation. One of them checks for number in words, basically in a morphological way, but exceptions like "Buenos Aires" and "people" must also be dealt with. Associated routines are capable of forming plurals, or constructing singulars from given plurals (incidentally, these functions are also used by the retrieval functions and other components of SCHOLAR). Other auxiliary routines are POS (for "part-of-speech") which checks the part of speech of a word or an atomic list, and 1STPOS ("first part-of-speech") which extracts from a complex list the first word which is a given part-of-speech. This last function is important in the operation of INT-E-Ø, INT-E-PRED, and it is also used by other components of SCHOLAR like E-3 when analyzing student input. Still other auxiliary routines, like DET and A-AN, handle the assignment of determiners.

So far we have presented some examples of output produced by SCHOLAR's English-test generator, and discussed this module in a structural way. Let us now analyze it in a dynamic way.

In general terms, the English-text generator is a highly recursive, single-pass set of routines, with look-ahead and look-behind capabilities that make the above routines context-sensitive; we think that the best way to further discuss this is by showing (see Fig. A.13, a to e) a trace of the English-text generator operating on an actual example. In this trace, some of the most important procedures originate a type-out of their list of arguments, and of the value they return. These type-outs appear interleaved with actual text being produced by the text generator. Indentations are automatically made in the tracings in relation to the level of embedding.

We see in Fig. A.13 (a) the question which will be answered with the last portion of the answer of Fig. A.13, and then E-3 returning a list of information. This is the first argument X for INT-E-Ø (the call to INT-E is not traced), the second, ATTØ, being the name of the requested property. The argument N is related to indentation. The argument NODE is T if X is a node in the semantic network and NIL if it is a value, argument A is related to determiners, PAST is obvious, and AA can refer to some adverbial modifiers. The next call to INT-E-Ø selects the first element from the former value, with NODE = T. Now INT-E-Ø finds no noun in X (neither CN or XN), and decides to consider X as the predicate of a sentence, by calling INT-E-SENT. This procedure first calls ATTØ which modifies nothing. Then VRB returns "(ARE)". In (b) OF-IN-ON converts (PLAINS ARGENTINA) into (THE PLAINS OF ARGENTINA) which is then typed out, together with the verb. Then INT-E-PRED recursively analyzes the predicate, and produces the type-out "USUALLY FERTILE."

Control is then returned to INT-E-Ø which proceeds with the second element of the original value of X. Processing is similar to that in the first case, with some variations. One of them is the action of ATTØ which now eliminates "Argentina" from the future subject of the sentence, since its presence would be redundant if we take into account the former sentence, already typed out. Next we see the VRB returns (IS) in spite of having "plains" in the subject. This is because the predicate is singular. When the proposed subject is shown again, as an argument to OF-IN-ON, we observe that VRB has properly changed "plains" into "plain." OF-IN-ON in turn correctly processes the conjunctive adjective ($L EASTERN CENTRAL), and the correct sentence "The eastern and central plain is the Pampa" is formed.

Control is then returned again to INT-E-Ø which starts processing the last element of the original value X. The tentative subject proposed by INT-E-Ø to INT E SENT (see (d) in Fig. A.13) is now

```
←Q/A)

*TELL ME ABOUT THE PLAINS IN ARGENTINA.*

E-3:
INPUT = (TELL\ME ABOUT THE PLAINS IN ARGENTINA ".")
NBDØ = T
MODE = NIL

E-3 = ((FERTILE NIL USUALLY) (($L EASTERN CENTRAL)
NIL PAMPA) (NORTHERN NIL CHACO))


INT-E-Ø:
X = ((FERTILE NIL USUALLY) (($L EASTERN CENTRAL) NIL
PAMPA) (NORTHERN NIL CHACO))
ATTØ = (PLAINS ARGENTINA)
N = Ø
NODE = NIL
A = NIL
PAST = NIL
AA = NIL


   INT-E-Ø:
   X = (FERTILE NIL USUALLY)
   ATTØ = (PLAINS ARGENTINA)
   N = Ø
   NODE = T
   A = NIL
   PAST = NIL
   AA = NIL


      INT-E-SENT:
      PRED = (FERTILE NIL USUALLY)
      ATTØ = (PLAINS ARGENTINA)
      A = NIL
      AA = NIL
      N = 2
      BB = T
      PAST = NIL
      PUNCT = NIL
      VRB = NIL


         ATTØ:
         ATTØ = (PLAINS ARGENTINA)
         PRED = (FERTILE NIL USUALLY)

         ATTØ = (PLAINS ARGENTINA)

         VRB:
         ATTØ = (PLAINS ARGENTINA)
         PRED' = (FERTILE NIL USUALLY)
         PAST = NIL

         VRB = (ARE)
```

Fig. A.13 (a) Traced Protocol of an Example of English-Text Generation (CONT)

```
        OF-IN-ON:
        X = (PLAINS ARGENTINA)
        A = NIL
        AFTXN = NIL

        OF-IN-ON = (THE PLAINS IN ARGENTINA)
THE PLAINS IN ARGENTINA ARE
        INT-E-PRED:
        Y = (FERTILE NIL USUALLY)
        NODE = T
        A = NIL
        XCN = NIL
        AB = NIL


           INT-E-PRED:
           Y = (USUALLY)
           NODE = NIL
           A = NIL
           XCN = NIL
           AB = (FERTILE)

USUALLY FERTILE                INT-E-PRED = 16
        INT-E-PRED = NIL

     INT-E-SENT = NIL
  INT-E-Ø = T

  INT-E-Ø:
  X = (($L EASTERN CENTRAL) NIL PAMPA)
  ATTØ = NIL
  N = Ø
  NODE = T
  A = NIL
  PAST = NIL
  AA = NIL


     INT-E-Ø:
     X = (PAMPA)
     ATTØ = (($L EASTERN CENTRAL) PLAINS ARGENTINA)
     N = 2
     NODE = NIL
     A = NIL
     PAST = NIL
     AA = NIL


        INT-E-SENT:
        PRED = (PAMPA)
        ATTØ = (($L EASTERN CENTRAL) PLAINS ARGENTINA)
        A = NIL
        AA = NIL
        N = 2
        BB = NIL
        PAST = NIL
        PUNCT = NIL
        VRB = NIL
```

Fig. A.13 (b) Traced Protocol of an Example of English-Text Generation (cont.)

```
            ATTØ:
            ATTØ = ((SL EASTERN CENTRAL) PLAINS ARGENTINA)
            PRED = (PAMPA)

            ATTØ = ((SL EASTERN CENTRAL) PLAINS)

            VRB:
            ATTØ = ((SL EASTERN CENTRAL) PLAINS)
            PRED' = (PAMPA)
            PAST = NIL

            VRB = (IS)

            OF-IN-ON:
            X = ((SL EASTERN CENTRAL) PLAIN)
            A = NIL
            AFTXN = NIL

            OF-IN-ON = (THE EASTERN AND CENTRAL PLAIN)
THE EASTERN AND CENTRAL PLAIN IS
            INT-E-PRED:
            Y = (PAMPA)
            NODE = NIL
            A = NIL
            XCN = NIL
            AB = NIL

THE PAMPA             INT-E-PRED = 10

         INT-E-SENT = NIL

         INT-E-Ø:
         X = NIL
         ATTØ = NIL
         N = 2
         NODE = NIL
         A = NIL
         PAST = NIL
         AA = NIL

         INT-E-Ø = T
      INT-E-Ø = T
   INT-E-Ø = T

   INT-E-Ø:
   X = (NORTHERN NIL CHACO)
   ATTØ = NIL
   N = Ø
   NODE = T
   A = NIL
   PAST = NIL
   AA = NIL
```

Fig. A.13 (c) Traced Protocol of an Example of English-Text Generation (cont.)

```
INT-E-Ø:
X = (CHACO)
ATTØ = (NORTHERN ($L EASTERN CENTRAL) PLAIN)
N = 2
NODE = NIL
A = NIL
PAST = NIL
AA = NIL

    INT-E-SENT:
    PRED = (CHACO)
    ATTØ = (NORTHERN ($L EASTERN CENTRAL) PLAIN)
    A = NIL
    AA = NIL
    N = 2
    BB = NIL
    PAST = NIL
    PUNCT = NIL
    VRB = NIL

        ATTØ:
        ATTØ = (NORTHERN ($L EASTERN CENTRAL) PLAIN)
        PRED = (CHACO)

        ATTØ = (NORTHERN PLAIN)

        VRB:
        ATTØ = (NORTHERN PLAIN)
        PRED' = (CHACO)
        PAST = NIL

        VRB = (IS)

        OF-IN-ON:
        X = (NORTHERN PLAIN)
        A = NIL
        AFTXN = NIL

        OF-IN-ON = (THE NORTHERN PLAIN)
THE NORTHERN PLAIN IS
        INT-E-PRED:
        Y = (CHACO)
        NODE = NIL
        A = NIL
        XCN = NIL
        AB = NIL
```

Fig. A.13 (d) Traced Protocol of an Example of English-Text Generation (cont.)

```
 THE CHACO               INT-E-PRED = 10
.
         INT-E-SENT = NIL

         INT-E-0:
         X = NIL
         ATT0 = NIL
         N = 2
         NODE = NIL
         A = NIL
         PAST = NIL
         AA = NIL

         INT-E-0 = T
      INT-E-0 = T
   INT-E-0 = T
INT-E-0 = T

*

←
```

Fig. A.13 (e) Traced Protocol of an Example of English-Text Generation (CONCLUDED) an incorrect form, (NORTHERN ($L EASTERN CENTRAL) PLAIN) obtained by concatenation of the adjective "northern" with the previously proposed subject. But the procedure ATTØ performs the necessary filtering; it recognizes the old adjective, and eliminates it, returning "northern plain." The rest of the traced protocol merits no further comments.

Though the traced protocol of Fig. A.13 which we have discussed above does not present all the capabilities of the SCHOLAR English-text generator, it does illustrate, within constraints of space, its general characteristics.

A.7 Overall Question-Answering Procedures

In a gross division of tasks in SCHOLAR, question-answering procedures are one of the major modules. In the last three subsections, we have independently seen the three separate components (retrieval procedures, student-input processing, and English-text generation) whose aggregate would essentially represent the question-answering part of SCHOLAR. We think, however, that it is convenient to give the reader a brief unified view of SCHOLAR's question-answering capabilities. This will be done in this subsection by presenting and discussing examples.

Figure A.14 presents an on-line protocol with SCHOLAR operating in its question-answering mode. In the first question, information about the number of countries is not explicitly present as such in the data base, so SCHOLAR counts the countries to produce an answer. All the following questions but the last are various ways to ask about the population size of Venezuela. This gives an idea of the relative freedom permitted in formulating a question. Observe the "How many people . . .?" question, where population is not mentioned, and the alternative requests for either "population," "size of the population," and "population size" in questions 3 to 5. Question 6 is a somewhat awkward construction which SCHOLAR manages, however, to comprehend. Questions 7 and 8 are two forms of a true-false question on the same subject as the former questions; note that question 8 could also be called a yes-no question. Observe the different answers to questions 7 and 8, due to the different values proposed by the student; since SCHOLAR is using its general matching routines in answering true-false questions, it can accept an approximate statement. Question 9, again on the population of Venezuela, is an example of fill-in question which the system properly "comprehends," and then answers. Finally, question 10 was included because of the nice way in which OF-IN-ON handled a long string as subject of the answer sentence; the explicative (which could be omitted) is written as such, between commas, since the previous noun is an XN, which together with the

```
Q/A)

*HOW MANY COUNTRIES ARE THERE IN SOUTH AMERICA?*

   THE COUNTRIES OF SOUTH AMERICA ARE 11.

*HOW MANY PEOPLE ARE THERE IN VENEZUELA?*

   THE POPULATION IS APPROX 8000000. PEOPLE.

*WHAT IS THE POPULATION IN VENEZUELA?*

   THE POPULATION IN VENEZUELA IS APPROX 8000000. PEOPLE.

*
 WELL, HAVE YOU MADE UP YOUR MIND?

*WHAT IS THE SIZE OF THE POPULAAT↑T↑ATIION↑N↑O↑ION IN
VENEZUELA?*

   THE POPULATION IN VENEZUELA IS APPROX 8000000. PEOPLE.

*GIVE ME THE POPULATION SIZE OF VENEZUELA.*

   THE POPULATION IN VENEZUELA IS APPROX 8000000. PEOPLE.

*HOW MUCH POPULATION HAS VENEZUELA?*

   THE POPULATION IN VENEZUELA IS APPROX 8000000. PEOPLE.

*TELL ME IF THE POPULATION IN VENEZUELA IS 7000000.*
 THAT'S ABOUT RIGHT. THE EXACT ANSWER IS 8000000..

*IS THE POPULATION OF VENEZUELA 3000000?*
 WRONG.
 THE CORRECT ANSWER IS: 8000000.

*
 YOU ARE TAKING TOO MUCH TIME...

*THE POPULATION OF VENEZUELA IS ---.*

   THE POPULATION IN VENEZUELA IS APPROX 8000000. PEOPLE.

*WHAT IS THE RELATIONSHIP BETWEEN THE ACONCAGUA AND ARGENTINA?*

   THE HIGHEST ALTITUDE OF THE ANDES, THE PRINCIPAL
       MOUNTAINS IN ARGENTINA, IS THE ACONCAGUA.
```

Fig. A.14 On-Line Protocol Using SCHOLAR's Question-Answering Mode initial part would suffice for naming purposes; observe that answering this question has involved, through RET, a call to RETA with the retrieval of a very long attribute chain.

Figure A.15 is essentially a display of different possible forms and stylistic variations for the formulation of true-false questions. The response about Rio de Janeiro occurred because SCHOLAR had it undefined at the time when the protocol was taken. Observe that the two last questions are of a slightly different nature, since we are there asking about superconcepts.

Figure A.16 is a protocol of the process of comprehending and answering the second question presented in Fig. A.14 above. In Fig. A.16 (as in the previous subsection's Fig. A.13) some of the most important routines involved have been traced, i.e., their calls and returns are printed-out, and appear mixed with ordinary student and SCHOLAR typing. Incidentally, the way question 1 of Fig. A.14 is processed is very different from the procedure displayed here. Going back to Fig. A.16, "people" is not an attribute of any property in Venezuela. But SCHOLAR recognizes that "how many" is asking for a number. SCHOLAR examines the information associated with "people" and discovers that it can be considered as a unit (in the numerical sense) applied to "population." It then searchs for the population of Venezuela, and within it, retrieves the top numerical property. Having printed out the number 8000000. it wants to assign a unit to it. By now, SCHOLAR has forgotten that in processing the question it started out with "people." So it searches the subject of the sentence, extracts the first concept noun in it, and investigates as to whether that CN has a unit (again in the numerical sense). This way SCHOLAR rediscovers that the "unit" of "population" is "people," and prints this word out.

A.8 Question Generation and Context-Handling Procedures

In this subsection we will discuss how SCHOLAR handles the generation of questions and, when necessary, of contexts. Part of the latter operation was described in Subsection A.2c, when we talked about the context push-down list CTXPDL.

The fundamental function here is QGEN (for "question generation). The principal arguments are a context X, and a mode of questioning MOD. But one or both of them may be missing, in which case, QGEN properly generates them.

```
Q/A)

*TELL ME IF MONTEVIDEO IS THE CAPITAL OF URUGUAY.*
 VERY GOOD.


*TELL ME IF THE CAPITAL OF URUGUAY IS BUENOS AIRES*

 WHY ARE YOU TAKING SO MUCH TIME? NOW, ABOUT YOUR STATEMENT...
 YOUR ANSWER IS INCORRECT.
 THE CORRECT ANSWER IS: MONTEVIDEO


*IS MONTEVIDEO THE CAPITAL OF URUGUAY?*
 YOU ARE CORRECT.


*IS THE CAPITAL OF URUGUAY MONTEVIDEO?*
 THAT'S FINE.


*IS IT TRUE THAT THE CAPITAL OF URUGUAY IS MONTEVIDEO?*
 THAT'S RIGHT.


*IS IT TRUE THAT RIO DE JANEIRO IS THE CAPITAL OF URUGUAY?*

 IN WHAT YOU SAY, I DON'T RECOGNIZE THE WORD "RIO DE JANEIRO".
      AGAIN, PLEASE:

*IS IT TRUE THAT BUENOS AIRES IS THE CAPITAL OF URUGUAY?*
 THAT'S NOT RIGHT.
 THE CORRECT ANSWER IS: MONTEVIDEO


*IS 60 THE AVERAGE TEMPERATURE IN MONTEVIDEO?*
 WHAT YOU SAY IS APPROXIMATELY CORRECT. BUT A MORE EXACT
      VALUE IS 65.


*IS PERU A COUNTRY?*
 VERY GOOD.


*IS MONTEVIDEO A CAPITAL?*
```

Fig. A.15 Handling of True-False Questions by SCHOLAR

```
 Q/A)

*HOW MANY PEOPLE ARE THERE IN VENEZUELA?*

E-3:
INPUT = (HOW\MANY PEOPLE ARE\THERE IN VENEZUELA ?)
NBDØ = T
MODE = NIL


   RETV:
   ATT = SUPERC
   OBJ = (PEOPLE)
   FLGQ = NIL
   AA = NIL

   RETV = ((UNIT))

   RETV:
   ATT = APPLIED\TO
   OBJ = (PEOPLE)
   FLGQ = NIL
   AA = NIL

   RETV = ((POPULATION))

   RETV:
   ATT = POPULATION
   OBJ = VENEZUELA
   FLGQ = NIL
   AA = NIL

   RETV = (((APPROX NIL \ 8ØØØØØØ.)))
E-3 = ((APPROX NIL \ 8ØØØØØØ.))

INT-E:
X = ((APPROX NIL \ 8ØØØØØØ.))
ATTØ = POPULATION
N = Ø
NODE = NIL
A = NIL
PAST = NIL
AA = NIL
BB = T

   THE POPULATION IS APPROX 8ØØØØØØ.
   RETV:
   ATT = UNIT
   OBJ = (APPROX NIL \ 8ØØØØØØ.)
   FLGQ = NIL
   AA = NIL

   RETV = NIL

   RETV:
   ATT = UNIT
   OBJ = POPULATION
   FLGQ = NIL
   AA = NIL

   RETV = ((PEOPLE))
  PEOPLE.
INT-E = T

*
```

Fig. A.16 Traced Protocol of an Example of Question-Answering by SCHOLAR

A basic routine is TETGEN (for "retrieve and generate"), which when given a type node in the information structue, selects through recursive calls an appropriate node pointed by the corresponding unit. If we are trying to select a subcontext within a context, RETGEN is called by CTXGEN with the argument CTXGEN = T. If the purpose is to select a string from which to form a question, RETGEN tries to select an appropriate terminal node (as said before, this facilitates handling the student answer). An argument called CHNFLG controls the possibility of jumping from the initial unit to another unit; a question like "What is the population size of the capital of Uruguay?" is originated this way. Another argument, BRK$, controls the possibility of breaking a list of elements and requesting only one of them in the question, as is the case in "Is Cordoba one of the cities in Argentina?"

The effect of both CHNFLG and BRK$ upon RETGEN is probabilistic. In the present version of SCHOLAR, the selection of each element added to a partially formed string is also probabilistic, with weights which inversely depend on the irrelevancy tags. Finally, withing RETGEN, temporary tags are properly handled to keep track of what has been already asked (see also subsection A.2d).

The procedure that selects a mode from available lists of modes in which weights are given to them, is called MODGEN (for "mode generation") and is also probabilistic. This procedure is usually called by STR-Q (for "string-to-question") which is the top routine in the group handling the conversion of a string into an English sentence. STR-Q calls other routines like STR-A ("string-to-affirmative"), STR-WH ("string-to-WH/question") which in turn can call SPATT-WH ("special attribute for WH questions"). The general INT-E-SENT can be called in several cases, for example, by STR-A.

In SCHOLAR's question generation procedures the questioned element is always the value of the generalized triple. Within this, SCHOLAR is capable of generating questions of four basic types: WH-questions, true-false, fill-in, and multiple-choice, with many possible variations in each. Of course, multiple-choice questions require the formulation of a question for which alternative answers are given; that question can be a WH or fill-in question. Multiple-choice and "incorrect" true-false questions require the generation of alternatives to the correct statement. This is done by the procedure called ALTGEN which has three arguments. The first is the value which we want to replace, the second is the required number of alternatives, and the third is the probability of generating alternatives unrelated to the given value, and alternatives being considered related if it has the same superconcept as the value;

even unrelated alternatives are the same part-of-speech words as the value they could replace. The number of alternatives is currently set to three in multiple-choice, which, including the correct value, gives the student four choices. The number of alternatives to generate is, of course, only one for wrong true-false questions.

When the value is numerical, be it a single number or a range, ALTGEN calls another routine called ALT#GEN. It generates the proper single numbers or numerical ranges, with each generated number being within a certain multiplicative range of the corresponding correct number, but also different enough (at least by a factor of 2) from it to avoid considering the alternative as correct.

It must be said that selection of a question, and especially, of the syntax for it, it is a matter of experimentation, convenience, and even taste, For example, in SCHOLAR there are some strings which only yield true-false or multiple-choice questions. This is the case, for example, with strings which have an adjective as a value. If an open question is asked from the string "varied topography Argentina," many different correct answers could be given instead of "varied." For these reasons, we are using true-false questions, and sometimes multiple-choice questions as a catch-all category. This might suggest that we could (or should) use true-false questions less often in other cases where other forms are possible.

Another more or less arbitrary choice in SCHOLAR is the assignment of equal probability to correct and wrong statements in true-false questions. The wrong statements are originated by replacing the questioned value with an alternative of the same kind (which we have defined above as that having the same superconcept). If this cannot be found, perhaps because the superconcept is undefined, SCHOLAR currently forms the negation of the original statement ("Is it true that the topography in Argentina is <u>not</u> varied?"). This sometimes originates questions that are somewhat bizarre, and another strategy might be preferable.

Still another questionable decision applies to multiple-choice questions. Quite often, multiple-choice questions generated by human teachers contain some unreasonable items, i.e. items that are unrelated to the one they could replace. We have experimented with this, and most of the protocols have been taken with a fifty percent probability of originating unrelated alternatives. This seems too high now; that figure should either be zero, or a low number.

The discussion above illustrates the modularity and adjustability of SCHOLAR, and evidences its potential value to test strategies in verbal communication and teaching.

Let us now look at the dynamics of question generation. Figure A.17 is essentially a traced protocol of that. First, QUESLIS and CTXPDL are checked. The latter is at its initial stage, which means that a subcontext will have to be generated. Then a call is made to QGEN; no arguments are given which means that the routine will have to provide both its context and mode. And, effectively, QGEN calls CTXGEN with argument "South America," the overall context in order to generate a suitable subcontext. The subcontext is selected by means of a call to RETGEN with object "South America" and CTXGEN = T. The context "Venezuela" is obtained. Next, QGEN calls RETGEN again, this time with object "Venezuela" and CTXGEN = NIL; the latter means that a question-string is sought. RETGEN returns a string and no mode, and the QGEN calls STR-Q without specifying a mode. This implies a call to MODGEN which probabilistically returns T/F (true-false). With this, STR-Q decides to present an incorrect true-false question, selects one style of true-false presentation, and forms the sentence by calling STR-A which in turn calls INT-E-SENT (these last steps are not shown in the traced protocol). Observe that STR-Q returns the used mode as its value. Finally, QGEN appends the new context to CTXPDL and the new question to both CTXPDL and QUESLIS, and returns the current state of the latter.

Figure A.18 shows an exhaustive generation of strings out of a given unit. This is done by means of an auxiliary function, PRUEBA, which repetitively calls RETGEN with first argument CHILE, till RETGEN returns NIL indicating there are no more possible strings available. We are also showing in this figure the internal representation of the unit CHILE. Observe that the property with attribute superp does not originate any string since the irrelevancy tag is 6, which is a way of giving it zero relevancy. Effectively, we do not want a question from it since it overlaps with "location." Also observe that the strings return the chain object-attributes-value in reverse order; the purpose is to facilitate the construction of the subject in the questions which usually take attributes in an order opposite to that in the object tree-list.

Figure A.19 presents the result of different calls to STR-Q with different modes but the same string. The first multiple-choice question was originated with a fifty percent probability of unrelated alternatives, yielding only one city together with an ocean

```
←E QUESLIS
NIL
←E CTXPDL
((SOUTH\AMERICA (DURØ 3600 DI Ø LIF 7207)))
←QGEN()

QGEN:
X = NIL
MOD = NIL
BLK = NIL
C = NIL


   CTXGEN:
   CTXØ = SOUTH\AMERICA


      RETGEN:
      X = SOUTH\AMERICA
      CHNFLG = Ø
      BRK$ = Ø
      CTXGEN = T
      NOTATT = NIL
      A = NIL

      RETGEN = VENEZUELA
   CTXGEN = VENEZUELA

   RETGEN:
   X = VENEZUELA
   CHNFLG = Ø
   BRK$ = Ø
   CTXGEN = NIL
   NOTATT = NIL
   A = NIL

   RETGEN = (8000000. APPROX POPULATION VENEZUELA)

   STR-Q:
   STR = (8000000. APPROX POPULATION VENEZUELA)
   CTX = VENEZUELA
   MOD = NIL
   PAST = NIL
   C = NIL


      MODGEN:
      VAL = (8000000.)
      Y = ((MCH (I 3)) (FILL-IN (I 2)) (WH) (T/F (I 4)) (TRANSFO
(I 6)) (EXAMPLE (I 6)))
      Y' = ((MCH (I 3)) (FILL-IN (I 3)) (WH (I 3)) (T/F (I 3)))

      MODGEN = T/F
 PLEASE INDICATE IF THE FOLLOWING STATEMENT IS CORRECT
      OR INCORRECT:

 THE POPULATION IN VENEZUELA IS APPROX 2102784. PEOPLE.
   STR-Q = T/F
QGEN = (((I SCHOLAR 1) (CTX VENEZUELA QUESINTERRL (1) #QUES 1 DI
5) (((8000000. APPROX POPULATION VENEZUELA UNIT UNIT) (APPROX POPULATION
VENEZUELA)) VENEZUELA T/F) ((8000000. APPROX POPULATION VENEZUELA
UNIT UNIT) T/F)))
```

Fig. A.17 Traced Protocol of Context and Question Generation by SCHOLAR

```
←PRUEBA (CHILE)

(SOUTH\AMERICA LOCATION CHILE)
(214000. APPROX AREA CHILE)
(8500000. APPROX POPULATION CHILE)
(COUNTRY SUPERC CHILE)
(SPANISH LANGUAGE CHILE)
(SANTIAGO CAPITAL CHILE)
NILNIL
←



←EDITV (CHILE)
EDIT
*

PP

  (((XN CHILE)
      NIL)
    NIL
    (SUPERC NIL COUNTRY)
    (SUPERP (I 6)
      SOUTH\AMERICA)
    (AREA (I 2)
      (APPROX NIL \ 214000.))
    (LOCATION NIL SOUTH\AMERICA)
    (POPULATION NIL (APPROX NIL \ 8500000.))
    (LANGUAGE (I 2)
      SPANISH)
    (CAPITAL (I 1)
      SANTIAGO))
*OK
CHILE
←
```

Fig. A.18 Exhaustive Generation of Semantic Strings

```
←STR-Q((LIMA CAPITAL PERU) PERU WH)

   WHAT IS THE CAPITAL IN PERU?
WH
←

←STR-Q((LIMA CAPITAL PERU) PERU FILL-IN)

 THE CAPITAL IN PERU IS ---.
FILL-IN
←

←STR-Q((LIMA CAPITAL PERU) PERU MCH)


 USE ONE OF THE FOLLOWING:

 ANTARCTIC OCEAN
 LIMA
 GUYANA
 PUNTA DEL ESTE

 TO FILL THE BLANK BELOW:

 THE CAPITAL IN PERU IS ---.
FILL-IN
←

←EDITF(STR-Q)
EDIT
*(F ALTGEN T)
*P
(ALTGEN (CAR VAL) 3 50)
*(4 0)
*OK
STR-Q
←STR-Q((LIMA CAPITAL PERU) PERU MCH)


 SELECT AN ALTERNATIVE FROM THE LIST:

 SAO PAULO
 MONTEVIDEO
 LIMA
 CORDOBA

 FOR THE QUESTION:

   WHAT IS THE CAPITAL IN PERU?
WH
←

←STR-Q((P↑PLIMA CAPITAL PERU) PERU T/F)

 IS IT CORRECT TO SAY THAT THE CAPITAL IN PERU IS BRAZILIA?
T/F
←

←STR-Q((LIMA CAPITAL PERU) PERU T/F NIL T#↑#)

 THE CAPITAL IN PERU IS LIMA.
 TRUE OR FALSE?
T/F
←
```

Fig. A.19 Question Generation in Different Modes From a Given Semantic String and a country as alternatives for "Lima." This seemed bad, and through a quick editing, we set the probability of unrelated alternatives to zero, obtaining the second (and better, we believe) multiple-choice question. Finally, two true-false questions were generated. In the second, a fourth argument set to T assured us to form a "correct" true-false question.

In Fig. A.20 we have concentrated in showing the generation of Wh-questions from different strings. Observe "when" and the past tense in the first question, the latter originated by a tag in the property "history" of "Uruguay." Also observe that OF-IN-ON eliminates "history" from the answer, becuase "history" is implied by "independence," being its superpart.

When that elimination does not happen, bad questions are obtained. An example is the second one, originated by the fact that "war" was an undefined word. A quick definition of "war" with "history" as its superpart corrects the defect in the question when later formed again. Incidentally, observe that STR-Wh randomly selects between "where" and "in what" followed by superconcept. Other questions in Fig. A.20 show how superconcept and superpart strings are handled.

Finally, Figs. A.21 and A.22 respectively show a number of examples of generation of true-false and multiple-choice questions. Observe the variety of strings that can be handled, and the selection of styles available. Also observe that of the three multiple- choice questions in Fig. A.22, SCHOLAR decided to use a fill-in form in two of them, and a WH form in the third.

A.9 The Matching Routines and Error Handling

We have said in Section III of this work that in SCHOLAR we were adopting a matching technique to check students' answers. The top matching procedure, called MATCH1, compares the expect answer (EXPANS) generated together with the question, and the actual answer (ANS) given by the student. On the one hand, MATCH1 checks for interruptions; it also initially calls FLGQ, a routine which examines the expected answer and decides if it should be considered atomic, a list of elements, a number, or some other special case. The return of FLGQ is combined with system and question flags into a list of flags used in analyzing the student's response. Next MATCH1 compares EXPANS with ANS. If identical it returns the result as "perfect." If not, it finds the intersection between EXPANS and ANS, and also the non-common elements present in EXPANS and ANS. It also calls NBDBND (see Subsection A.5 above) with respect to unaccountable portions of the student answer. NBDBND returns the two

```
←STR-Q ((1825 DATE INDEPENDENCE HISTORY URUGUAY) URUGUAY WH)
 WHEN WAS THE INDEPENDENCE IN URUGUAY?
NIL
←

←STR-Q((1870 DATE PARAGUAYAN WAR HISTORY URUGUAY) URUGUAY WH)
 WHEN WAS THE PARAGUAYAN WAR OF HISTORY IN URUGUAY?
NIL
←

←SETQQ (WAR (((CN WAR)(DET THE))NIL (SUPERP NIL HISTORY)))
((& &) NIL (SUPERP NIL HISTORY))
←

←STR-Q((1870 DATE PARAGUAYAN WAR HISTOJY ↑ ↑Y↑JRX↑XY URUGUAY)
URUGUAY WH)
 IN WHAT DATE WAS THE PARAGUAYAN WAR IN URUGUAY?
NIL
←

←STR-Q((COUNTRY SUPERC CHILE) CHILE WH)
 WHAT IS CHILE?
NIL
←

←STR-Q((CLIMATE SUPERP TEMPERATURE) TEMPERATURE WH)
 WHAT IS A TEMPERATURE A PART OF?
NIL
←

←STR-O((SOUTH\AMERICA SUPERP ARGENTINA) ARGENTINA WH)
 WHAT CONTINENT IS ARGENTINA A PART OF?
NIL
←

←STR-Q(((SL SPAIN ITALY) PRINCIPAL COUNTRIES ORIGIN POPULATION
URUGUAY) URUGUAY WH)
   WHAT ARE THE PRINCIPAL COUNTRIES OF ORIGIN OF THE
POPULATION IN URUGUAY?
NIL
←


 STR-Q((SOUTH\AMERICA LOCATION ARGENTINA) ARGENTINA WH)
 IN WHAT CONTINENT IS ARGENTINA?
NIL
←

←STR-Q((URUGUAY LOCATION MONTEVIDEO) URUGUAY WH)
 WHERE IS MONTEVIDEO?
NIL
←
```

Fig. A.20 Formation of WH Questions From Different Strings

```
←STR-Q((SOUTH\AMERICA LOCATION URUGUAY) URUGUAY T/F)
 IS IT TRUE THAT URUGUAY IS LOCATED IN ASIA?
NIL
←

←STR-Q((SOUTH\AMERICA LOCATION URUGUAY) URUGUAY T/F)
 PLEASE INDICATE IF THE FOLLOWING STATEMENT IS CORRECT
OR INCORRECT:

 URUGUAY IS IN EUROPE.
NIL
←

←STR-Q((SOUTH\AMERICA LOCATION URUGUAY) URUGUAY T/F)
 IS IT CORRECT TO SAY THAT URUGUAY IS LOCATED IN ASIA?
NIL
←

←STR-Q(((30 90) RANGE TEMPERATURE CLIMATE URUGUAY) URUGUAY T/F)
 IS THE FOLLOWING TRUE OR FALSE?

 THE TEMPERATURE IN URUGUAY RANGES FROM 150 TO 450 DEGREES FAHRENHEIT.
NIL
←

←STR-Q((VARIED TOPOGRAPHY ARGENTINA) ARGENTINA T/F)
 IS IT TRUE THAT THE TOPOGRAPHY IN ARGENTINA IS VARIED?
NIL
←

←STR-Q((($L SPAIN ITALY) PRINCIPAL COUNTRIES ORIGIN POPULATION
URUGUAY) URUGUAY T/F)

 THE PRINCIPAL COUNTRIES OF ORIGIN OF THE POPULATION
      IN URUGUAY ARE SPAIN AND ITALY.
 RIGHT OR WRONG?
NIL
←
```

Fig. A.21 Generation of Various True-False Questions by SCHOLAR

```
←STR-Q(((30 90) RANGE TEMPERATURE CLIMATE URUGA↑AUAY) URUGUAY MCH)

 USE ONE OF THE FOLLOWING:

 6 22
 15 360
 15 45
 30 90

 TO FILL THE BLANK BELOW:

      THE TEMPERATURE IN URUGUAY RANGES FROM --- TO ---
        DEGREES FAHRENHEIT.
NIL
←

←

←STR-Q((ARGENTINA WESTERN BOUNDARIES LOCATION URUGUAY) URUGUAY MCH)

 SELECT AN ALTERNATIVE FROM THE LIST:

 CORDOBA
 ACONCAGUA
 PUNTA DEL ESTE
 ARGENTINA

 TO COMPLETE THE SENTENCE:

      THE WESTERN BOUNDARY IN URUGUAY IS ---.
NIL


←STR-Q((BUENOS\AIRES CAPITAL ARGENTINA) ARGENTINA MCH)

 SELECT AN ALTERNATIVE FROM THE LIST:

 PAYSANDU
 RIO DE JANEIRO
 BUENOS AIRES
 URUGUAY RIVER

 IN THE QUESTION:

    WHAT IS THE CAPITAL IN ARGENTINA?
NIL
←
```

Fig. A.22 Generation of Various Multiple-Choice Questions by SCHOLAR lists of bound and unbound atoms in the student answer. Bound atoms can later be investigated for misconceptions.

Next, and depending on the type of expected answer, MATCH1 calls routines like ATOMMATCH (matching atoms), LISTMATCH matching lists), or #MATCH (matching numbers). Finally, MATCH 1 returns a list formed by the following elements:

(1) a word characterizing the degree of matching; this can be one of the following: perfect, correct (correct but not identical), correct (used for lists when enough but not all elements are given by the student), missing (no wrong elements, but some expected ones are missing), wrong, approximately\correct (used for numbers), partc/partw (for partially correct - partially wrong), toomuch (extraneous elements added to an otherwise correct list).

(2) expected answer EXPANS.

(3) actual student answer ANS.

(4) intersection between (2) and (3).

(5) result of removing (4) from (2).

(6) result of removing (4) from (3).

(7) list of bound atoms in (6).

(8) list of unbound atoms in (6).

(9) list of current flags.

This list of items gives a fairly comprehensive picture about the student's answer and permits consequent decisions.

Let us now look at how atoms and lists are handled by the matching procedures. Figure A.23 presents an on-line testing of MATCH1, with different possible arguments. Misspellings are handled for both single-atom responses and multi-atom ones. The current misspelling routine in SCHOLAR is based in a percentage of correct letters which is a system parameter; it is currently set at 70 percent. Observe the "partc/partw" case, and the list of lists returned by MATCH1. The number 3 indicates the length of the response which, if the atoms are correct, would make it acceptable. A system parameter acting into LISTMATCH can set that number to a percentage of the length of the expected answer. That parameter depends on

```
←MATCH1((ARGENTINA)(ARGENTINA))
(PERFECT (ARGENTINA) (ARGENTINA) (ARGENTINA) NIL
NIL NIL NIL (ATOM MISP APPROX))
←

←MATCH1((ARGENTINA)(URUGUAY))
(WRONG (ARGENTINA) (URUGUAY) NIL (ARGENTINA) (URUGUAY)
(URUGUAY) NIL (MISP APPROX))
←

←MATCH1((SPANISH)(SPANICH))
 I BELIEVE YOU MEANT TO TYPE "SPANISH".

(CORRECT (SPANISH) (SPANICH) NIL NIL NIL NIL NIL (MISP APPROX))
←

←MATCH1((URUGUAY ARGENTINA BRAZIL)(BRAZIL))
(MISSING (URUGUAY ARGENTINA BRAZIL) (BRAZIL) (BRAZIL) (URUGUAY
ARGENTINA) NIL NIL NIL (3 LIST MISP APPROX))
←

←MATCH1((URUGUAY ARGENTINA BRAZIL PARAGUAY)
           (URUGUAY ARGENTINA PARAGUAY))
(MISSING (URUGUAY ARGENTINA BRAZIL PARAGUAY) (URUGUAY ARGENTINA
PARAGUAY) (URUGUAY ARGENTINA PARAGUAY) (BRAZIL) NIL NIL NIL (4
LIST MISP APPROX))
←

←MATCH1((URUGUAY ARGENTINA  BRAZIL)
            (URAGUAY ARGENTINA COLOMBIA PERU))
 YOU MISPELLED "URUGUAY".

(PARTC/PARTW (URUGUAY ARGENTINA BRAZIL) (URAGUAY ARGENTINA COLOMBIA
PERU) (ARGENTINA URUGUAY) (BRAZIL) (COLOMBIA PERU) (COLOMBIA PERU)
NIL (3 LIST MISP APPROX))
←

←MATCH1((URUGUAY ARGENTINA)(COLOMBIA PERU))
(WRONG (URUGUAY ARGENTINA) (COLOMBIA PERU) NIL (URUGUAY ARGENTINA)
(COLOMBIA PERU) (COLOMBIA PERU) NIL (2 LIST MISP APPROX))
←
```

Fig. A.23 Matching Expected Vs. Student Answers (Literal) flags; if "some" appears in the question and then as a flag, only a low percentage of the items will be requested; if "all," then all of them should be given; it "one," only one.

Unfortunately, no utilization of synonyms is made in Fig. A.23. But SCHOLAR would accept them; for example, it considers correct the answer "U.S." if the expected answer is "United States," or "height" if the expected anser is "altitude."

Figure A.24 presents an on-line testing of the matching routines in the case of numerical answers. We see that the program accepts both exact and approximate numerical answers (the flag APPROX is on). The acceptance of approximate answers depends on a system parameter, #AP, currently set at 1.3. If ANS is between EXPANS divided by 1.3 and EXPANS times 1.3, then ANS is accepted as approximately correct.

For numerical ranges, in which ANS and EXPANS are pairs of numbers, #MATCH examines the lower number, the higher one, and the difference between them. Each of these three numbers obtained from ANS must be approximately correct in order for ANS itself to be considered as approximately correct.

The output from MATCH1 is taken by the procedure NEXT, which can call the more specialized routines T/FMESS and BRANCH. NEXT and T/MESS are responsible for typing appropriate messages to the student, some of them of a more or less constructed form (for example, in the case of partially correct-partially wrong answers). BRANCH is a very important compoent in an ISO CAI system. We have explained before, however, that in our developmental effort study of program actions conditional on student's errors had to be postponed till other more elementary components of SCHOLAR were ready. Because of this, BRANCH is not as developed as other parts of the system. It is set, however, to have certain interesting capabilities. For example, in numerical wrong answers, BRANCH can check if the answer is completely unreasonable in terms of a range defined in the semantic network for the attribute to which the number applies. For example, in no place on earth the temperature averages, say, 150 degrees Fahrenheit. A student's response, say 350, for an average temperature with correct value 50, is much worse than a response of, say, 80. Those unreasonable values could trigger further actions by the computer.

In the case of symbolic answers, if a student asked about the capital of Argentina responds Brazilia, he is not making as serious a mistake as that made if he would answer Brazil (which is a country). When an atomic response is unreasonable (superconcept

```
←MATCH1((25)(25))
(PERFECT (25) (25) (25) NIL NIL NIL NIL (# MISP APPROX))
←

←MATCH1((24000000.)(24000000.))
(PERFECT (24000000.) (24000000.) (24000000.) NIL NIL NIL NIL (#
MISP APPROX))
←

←MATCH1((50)(55))
(APPROXIMATELY\CORRECT (50) (55) NIL (50) (55) (55) NIL (# MISP
APPROX))
←(MATCH1#
MATCH1((50 )(80))
(WRONG (50) (80) NIL (50) (80) (80) NIL (# MISP APPROX))
←

←MATCH1((24000000.)(12000000.))
(WRONG (24000000.) (12000000.) NIL (24000000.) (12000000.) (12000000.)
NIL (# MISP APPROX))
←

←MATCH1((30 60)(30 60))
(PERFECT (30 60) (30 60) (30 60) NIL NIL NIL NIL (RANGE # MISP
APPROX))
←

←MATCH1((30 60)(32 65))
(APPROXIMATELY\CORRECT (30 60) (32 65) NIL (30 60) (32 65) (32
65) NIL (RANGE # MISP APPROX))
←

←MATCH1((30 60)(32 90))
(WRONG (30 60) (32 90) NIL (30 60) (32 90) (32 90) NIL (RANGE #
MISP APPROX))
←

←MATCH1((20 80)(10 80))
(WRONG (20 80) (10 80) (80) (20) (10) (10) NIL (RANGE # MISP APPROX))
←

←MATCH1((40 50)(35 55))
(WRONG (40 50) (35 55) NIL (40 50) (35 55) (35 55) NIL (RANGE #
MISP APPROX))
←
```

Fig. A.24 Matching Expected Vs. Student Answers (Numerical) different from that of the expected response), then an attempt is made by SCHOLAR to question the student on the superconcept of the wrong answers, and if that fails, on the location of that item. There are some other possibilities, like going to a "correct" true-false question, to request the student to try again, or finally, to give the correct answer to the student. These decisions are not probabilistic; a definite sequence is followed.

APPENDIX B

GENERAL APPROACH USED IN DEVELOPING SCHOLAR

B.1 Some Pedagogical Considerations

We will discuss here a series of questions that are of fundamental importance for the development of the better CAI systems we claim feasible and deserving immediate attention.

The first question is basically related to the different philosophy in ISO CAI versus the traditional AFO CAI. In the latter, the computer is given as its data base items that it must manipulate literally, with no latitude to draw inferences and generalizations. The computer in AFO CAI systems has no "knowledge"; it merely reproduces text, questions and answers that have been specifically entered in advance. It is unwise and even unfair to pretend that, under those conditions, the computer could even approximate the behavior of a human teacher. In his teaching process a human teacher is not reciting specific questions, but he is utilizing and processing knowledge he has stored in the form of a semantic information structure; experimental evidence (Collins and Quillian, 22) indicates that this information structure is a semantic network very much like that utilized in SCHOLAR. The important point is that when asking a question or responding to one put to him by a student, the teacher probes his own information storage and perhaps processes the result of that search to find a meaningful solution. Furthermore, this exploration requires the utilization of information not only about the specified subject on hand, but also about more general knowledge. For instance, asking a question about the average temperature in Montevideo and processing the corresponding student answer may require knowing that temperatures are measured in degrees Fahrenheit, and that nowhwere on earth is the average temperature above, say, 120 degrees Fahrenheit. Similarly, when we say that Brazil is larger than Argentina, we have the understanding that we are comparing the two countries in terms of their areas.

The development of SCHOLAR is a step in what we claim is the right direction, i.e. CAI programs that know what they are talking about, the same way human teachers do. This necessary preoccupation with properly representing and intelligently utilizing knowledge has led us to the use of a semantic network for the data base (see Section II.1) and generally, an artificial intelligence approach to the program. The generality of this approach makes the system adaptable for use as a decision aid as well as a CAI system.

The second point we want to make is related to some arguments presented above to support the first. Though we do not advocate the construction now of formal models of teachers' behavior (and SCHOLAR is not a modeling effort) we would like to argue that CAI can benefit from a close look at human teachers. Clearly, as we have said above, they do not act on the basis of stored questions, answers, etc., but on knowledge, in the form of facts, concepts, and procedures. Also, human teachers are not pre-programmed to the ultimate detail, as is, for example, the Socratic system (Swets and Feurzeig, 9) . They rather have general procedural guidelines and criteria which depend on their information structures and also on recent events (such as a student's response). Following Minsky, 23) we would like to argue against the notion that, in the absence of some rational criterion for decision, teachers decide because of their own "volition." As Minsky says, "...people are incapable of explaining how it (free will or volition) differs from stochastic caprice..." On the other hand, we do not have an understanding so complete as to explain all their acts on a rational, deterministic basis. In designing a program in the domain of artificial intelligence, we still often want to preserve the richness of variety of human decisions, even if we cannot or should not program all of them in their ultimate details. For these reasons, we have chosen, when no better decision rationale or heuristic is available, to make some decisions on a constrained weighted random basis (Carbonell, 24).

After writing the above paragraph, we have found that Ashby (25) is currently arguing in much the same way. He says: "In arriving at a decision, human judgment first should prevail; then chance should be used as the necessary supplement to bring the decision to uniqueness...Modern methods of decision-making use both, chance and human judgment. From this point of view the use of chance is in no way a 'denial of rationality.' On the contrary, chance is the intelligent man's method of selection when he knows that the quantity of information available to him as selector is less than the quantity of selecting demanded of him."

In SCHOLAR, this applies, for example, to the selection of questions within a given context. Our practical justification for using a weighted-random selection strategy is that, by doing so, the program's behavior looks richer and "more intelligent" than it does when questions are consistently selected by some deterministic (but perhaps equally arbitrary) criterion like ordering within the data base, or always presenting less weighted questions last (maybe then some topics would never be touched). On the other hand, it is clear that, this strategy not being a fundamental one, it could be very easily replaced by another, due to the modular structure of SCHOLAR.

A third point to discuss here is the importance of natural-language communication between man and computer. We think this is an important goal, and we have made a substantial effort to achieve dialogues in a rather large and comfortable subset of English. The results have been surprisingly good, as protocols illustrate. In spite of this, we consider that the importance of natural-language communication for CAI has been somewhat exaggerated. This is especially so in the case of researchers who have neglected other aspects to concentrate on natural language understanding. A case in point is the work by Simmons, Schwartz, and their associates (Schwartz et al., 26). In spite of using a semantic information storage (in the form of triples) they use it only for processing student's answers and, very recently, for generating true-false questions (Simmons, 27); other types of program questions, and answering questions from the student, have been left aside. Their efforts seem to be intermediate between AFO and ISO systems, with limited generative capabilities and no provision for mixed initiative. On the other hand, they can process ambiguous sentences, anaphoric references, etc. It seems to us that a more balanced assignment of priorities is shown by SCHOLAR.

A fourth point to discuss refers to the subject, much talked about in CAI, concerning the processing of unanticipated answers and the associated and frequently mentioned need to construct a model of the student. This will make it possible, it is argued, to process his errors, study his misconceptions, and take some remedial action about them (Taylor, 11). But that modeling task is not easy, and the great difficulty of constructing from scratch a model of each student has been a major stumbling block for many investigators in the area of computers and education.

We think that this difficulty can be surmounted by avoiding the incremental building of a model with starting point zero. Our approach is different: having the semantic network as an information structure on the subject being dealt with, it seems natural to consider it as an ideal input-output model of the ideal student. It is so to the extent that the semantic network, when interrogated, would give the same answers (namely, the correct ones) as a "perfect" student has his knowledge organized strictly the way the semantic network is, though work by Collins and Quillian (22) suggests that the discrepancy may be small; we simply claim that both would produce, when interrogated, essentially the same output.

What about other students, those that may give some erroneous answers? We will now make the plausible working hypothesis that we can still use the ideal model as a departing point, since, except in the most serious cases, errors in answers will be due to minor deviations in an information structure can produce quite noticeable differences in output. On the other hand, any substantial, massive deviations, apart from having a low likelihood in occurrence, would have a generalized and devastating effect upon output. A possible working assumption (yet to be tested) is then that a student's input-output behavior can be accounted for by introducing small perturbations in the semantic network, these perturbations being the means of modeling the student's errors.

This approach makes the modeling of a student much easier. We give him the benefit of the doubt and assume that he will be correct until proven wrong. The practical advantage is considerable, since we start with a complete model (the "ideal" structure) and hopefully, if the above working hypothesis is correct, this will be closer to a model of a real student than starting from zero. From a practical point of view, we need to be much less worried about modeling with our approach than with the classic and rather unsuccessful "building-from-scratch" approach. This is so because we admit deviations to exist only when errors (perturbations) have been detected and not yet corrected. If a correct answer is received, no modification is made on the model.

Before going into the next question, we must say that in our work on SCHOLAR so far we have not yet developed to their fullest extent the modeling diagnostic capabilities, since in a balance of priorities, the top one was to develop a working system for demonstration purposes.

B.2 Some Relevant Artificial Intelligence Questions

The goal of artificial intelligence research has been defined in the following way (Feigenbaum and Feldman, 28): "to construct computer programs which exhibit behavior that we call 'intelligent behavior' when we observe it in human beings." The development of SCHOLAR is, to a large extent, an effort in this direction, and can be legitimately considered to be in the field of artificial intelligence. We are referring now to some of SCHOLAR's capabilities like answering questions not specifically anticipated, constructing questions on given topics, and generally carrying on a mixed-initiative contextual dialogue with a human in a rather free and comfortable subset of English.

It would be difficult to detail all possible ways in which artificial-intelligence research has influenced the development of SCHOLAR. Suffice it to consider here the basic areas of research having a direct connection with particular portions of our work, and, on the other hand, some important attitudes and points of view pervading the development of SCHOLAR in general.

In terms of areas within artificial intelligence, the most important for us is that related to knowledge structures, which are the essential basis for the ISO approach. In this sense, work on semantic information structures is highly relevant, and Minsky's collection (Minsky, 23) is an important reference. But we have been specially influenced by the work of Quillian on semantic networks, through both his writings (Quillian, 2,3) and invaluable personal discussions. It became clear early in our research that some form of a semantic network provided the kind of data base capable of supporting an ISO CAI system with the general characteristics that SCHOLAR now exhibits. Our network has characteristics different from Quillian's because of the rather different areas of application. But by and large, our data structure is largely inspired in Quillian's work.

The second area we would like to mention is that of natural language communication with computers. Here the work of investigators like Bobrow (29), Quillian again (3), and Simmons (30) has been of special interest to us. These investigators have adopted an approach in which both syntax and semantics play an important role in language comprehension. They have tended to emphasize (correctly, we believe) the semantic aspects, i.e., what questions and other statements mean rather than how they are structured. Considering semantics as an appendix to syntax is, we believe, basically incorrect; unfortunately, this widely-held point of view has led to the development of dozens of parsers and other syntax-oriented programs with limited practical consequences in general. This is, we believe, another area in which too much attention has been paid to form and too little to content (Minsky, 31).

The third area of artificial intelligence which directly relates to our work is that of question-answering systems. Question-answering systems have been investigated for a number of years. Interesting classical experimental systems are those by Green et al. (32), and Raphael (33). More recently, the work of Kellogg (34) is specially worth mentioning since his is a rather complete system with good data-base building facilities. It also has fairly interesting inferential capabilities (like comparing, counting, finding the largest element, etc.) used in question answering.

A very comprehensive review of natural-language question-answering systems has been made by Simmons(30), though the emphasis is probably more on natural-language analysis than on question answering. With the same approach, Schwarcz et al. (26) recently presented the last version of their series of Protosynthex systems for deductive question-answering using natural language. Some of the comments they make in this interesting paper are worth mentioning in relation to our own work on SCHOLAR.

They say first that "none of these systems [preceding theirs] is capable of expressing answers to retrieval requests in a flexible subset of English." They further say that some systems that return their output in a subset of English do so "through format matching and insertion rather than through linguistically motivated semantic analysis and generation procedures." Observe that SCHOLAR can express answers in a flexible subset of English, and does not use format insertion, but semantic retrieval and generation procedures.

Schwarcz and his associates further express in their conclusions that: "Another change that would be required to Protosynthex III into a question-answering system of practical utility would be the introduction of answering operators - such as count, list, name, and yes-no - and to allow some specification of the number of answers desired for the question (one, five, severa, all, etc.)." The reader will see, in Sections III and IV of the present work, that these problems have been generally taken into account in SCHOLAR, and most of them solved.

Finally, let us say that there is one new area of artificial intelligence in which we have found no antecedent for SCHOLAR. This is question generation and generally contextual mixed-initiative conversation. In this respect, SCHOLAR seems to be the first system of its kind.

Let us turn our attention now to some points of view currently held by some investigators in artificial intelligence. Some of these have had a strong influence on our approach to the development of SCHOLAR. They largely represent some points of view expressed by Minsky and the Artificial Intelligence Group at Project MAC, M.I.T. A good source in this respect is the Introduction by Minsky to the book on Semantic Information Networks edited by him (Minsky, 23).

About generality and knowledge, Minsky says that "the route toward generality must lie partly in more versatile organization of the knowledge-handling parts of the program's administration, and partly in the representation of more and better knowledge." Our development of ISO versus AFO CAI systems reflects, in part, that approach. Later on in the same article Minsky says: "I see no reason to believe that intelligence can exist apart from a highly organized body of knowledge, models, and processes." And still later: "The problem-solving abilities of a highly intelligent person lies in part in his superior heuristics for managing his knowledge structure and partly in the structure itself; these are probably somewhat inseparable." We are following this path when we emphasize the importance of semantic information structures for CAI, and the proper use of techniques for handling them.

Many investigators in early stages of artificial intelligence were very concerned about the learning capability of their programs. Some people, especially computer scientists only peripherally connected to artificial intelligence, still have that point of view. We have been asked repeated times: "How does SCHOLAR learn?" as if this were a sine-qua-non characteristic. Our standard response to that question is "SCHOLAR learns what is told." As Minsky puts it, "to make a machine with intelligence is not necessarily to make a machine that learns to be intelligent." And later: ". . . in our present state it will be more productive to try to understand how people understand so well what they are told than to focus exclusively on what they discover for themselves."

Finally, the following quote from the same source strictly represents our point of view in one aspect of the development of SCHOLAR: "But we have agreed to set aside the problem of acquiring knowledge [by a program] till we better understand how to represent and use it." In SCHOLAR we have postponed worrying about creation of the data base; we assume that it has been entered, that it exists. Our concern has been how to use and represent knowledge.

The problem of complexity represents another artificial-intelligence dimension where a substantial evolution has taken place. In early programs the approach was to define algorithms as simple as possible; the solution with the minimum number of rules and minimum amount of information was the most satisfactory one. This is still true, but investigators have come to the conclusion that in order to obtain complex behavior, programs may become complex, far from simple sequences of instruction. As Minsky puts it:..."the programmer will usually be unable to predict in advance all the details of ... [the components] interactions. For that, after all, is why he needs the computer."

Finally, let us conclude this section by saying a word on formality. It is usually intellectually pleasing to find algorithms with rules and schemata capable of describing complex behavior. But we must bear in mind that a working program is often as good a description as a formal mathematical or logical one. And, in many instances, an information-processing characterization is the only one possible. In SCHOLAR we have followed this approach. Instead of trying to develop a priori a theory of teaching, or learning, in a formal way, we have constructed a program that reasonably performs some conversation and teaching functions so far considered the privilege of human beings. We think that with the selected approach we have learned at least as much as with a more formal one.

APPENDIX C

SOME MORE SCHOLAR PROTOCOLS

```
←Q/A)
*HOW MANY COUNTRIES ARE THERE IN SOUTH AMERICA?*

   THERE ARE 11.

*H WOULD #
I WOULD LIKE TO KNOW SOMETHING ABOUT PERU*

   PERU IS A COUNTRY.
   IT IS LOCATED IN SOUTH AMERICA.
      THE CAPITAL IS LIMA.

*PLEASE TELL ME MORE ABOUT PERU.*

      THE AREA IS APPROX 480000.  SQUARE MILES.
      THE POPULATION IS APPROX 70000000. PEOPLE.

*
 PLEASE HURRY UP. I'M WAITING FOR YOUR STATEMENT.

*WHERE IS LIMA?*

   LIMA IS LOCATED IN CENTRAL PERU NEAR THE PACIFIC OCEAN.

*WHAT IS THE POPULATION OF LIMA?*

 SORRY, I DON'T KNOW.


*WOULD YOU PERHAPS KNOW THE CAPITAL OF ARGENTINA?*
 I DON'T UNDERSTAND WHAT YOU MEAN BY THE WORDS "PERHAPS" AND
"KNOW". AGAIN, PLEASE:

*WHAT IS THE CAPITAL OF ARGENTINA?*

   THE CAPITAL IN ARGENTINA IS BUENOS AIRES.

*HOW MANY PEOPLE ARE THERE IN ARGENTINA?*

   THE POPULATION IS APPROX 24000000. PEOPLE.
```

Fig. C.1 On-Line Protocol Using SCHOLAR's Question-Answering Mode

```
+++++++++++++++++++++++++++++++++++++++++++++++++++++++

 HELLO, MY NAME IS SCHOLAR. I WILL HELP YOU IN REVIEWING
      THE GEOGRAPHY OF SOUTH AMERICA. THIS IS ABC REGIONAL
      HIGH SCHOOL. TODAY IS 04/30/70.


 IF YOU HAVE CAREFULLY STUDIED YOUR INSTRUCTIONS, WE
      MAY BEGIN THE REVIEW NOW.

 WHEN YOU ARE READY TO START, TYPE READY --- REMEMBER
      TO TERMINATE WITH AN ASTERISK * FOLLOWED BY A
      CARRIAGE RETURN.

*REDY*

 NOT READY YET? LET ME KNOW AS SOON AS YOU CAN START.


*READY*

 PLEASE TYPE YOUR NAME --- MR., MISS, OR MRS. FOLLOWED
      BY FIRST NAME, MIDDLE NAME OR MIDDLE INITIAL,
      AND LAST NAME.

*MR. JONATHAN K. O'MALLEY*

 NAME OF YOUR INSTRUCTOR, PLEASE:

*MR. NORTHERLAND*


 LET'S BEGIN OUR DISCUSSION NOW, MR. O'MALLEY.

+++++++++++++++++++++++++++++++++++++++++++++++++++++++

 SELECT AN ALTERNATIVE FROM THE LIST:

 VENEZUELA
 URUGUAY RIVER
 EUROPE
 SOUTH AMERICA

 TO FILL THE BLANK BELOW:

 PARAGUAY IS LOCATED IN ---.

*VENEZUELA*
```

Fig. C.2 (a) On-Line Mixed-Initiative Protocol (CONT)

YOUR ANSWER IS INCORRECT.

IS THE FOLLOWING TRUE OR FALSE?

THE POPULATION IN PARAGUAY IS APPROX 1400000. PEOPLE.

TRUE

THAT'S RIGHT.

IS IT CORRECT TO SAY THAT THE CAPITAL IN PARAGUAY IS
NOT ASUNCION?

INCORRECT

VERY GOOD.

THE CAPITAL IN PARAGUAY IS ASUNCION.

USE ONE OF THE FOLLOWING:

MONTEVIDEO
EUROPE
PAMPA
SOUTH AMERICA

TO COMPLETE THE SENTENCE:

COLOMBIA IS IN ---.

SOUTH AMERICA

THAT'S FINE.

APPROX WHAT IS THE AREA OF COLOMBIA?

*
YOU ARE TAKING TOO MUCH TIME...

*
PLEASE HURRY UP. I'M WAITING FOR YOUR STATEMENT.

500000 SQUARE MILES.

APPROXIMATELY TRUE, BUT A MORE EXACT VALUE IS 482000.

COLOMBIA IS A ---.

CPUNTRY

YOU SHOULD HAVE WRITTEN "COUNTRY".

THAT'S RIGHT.

TYPE EITHER T OR F AFTER THE FOLLOWING STATEMENT:

THE POPULATION IN COLOMBIA IS APPROX 16000000. PEOPLE.

Fig. C.2 (b) On-Line Mixed-Initiative Protocol (cont.)

GIVE ME THE CAPITAL OF COLOMBIA?

THE CAPITAL IN COLOMBIA IS BOGOTA.

NOW ANSWER THE QUESTION YOU DIDN'T ANSWER BEFORE.

*COR
REC WELL, HAVE YOU MADE UP YOUR MIND?

CORRECT

REPHRASE YOUR STATEMENT. I CAN'T UNDERSTAND IT.

T

YOU ARE CORRECT.

THE LANGUAGE IN COLOMBIA IS ---.

Q/A

DO YOU WANT TO CHANGE THE GENERAL MODE OF THE INTERACTION
TO Q/A?
PLEASE ANSWER Y OR N.

Y

OK, MR. O'MALLEY. LET'S GO AHEAD.

PLEASE HURRY UP. I'M WAITING FOR YOUR STATEMENT.

WHAT IS THE AVERAGE TEMPERATURE IN THE CAPITAL OF URUGUAY?

THE AVERAGE TEMPERATURE IN THE CAPITAL IN URUGUAY
IS 65 DEGREES FAHRENHEIT.

Fig. C.2 (c) On-Line Mixed-Initiative Protocol (CONCLUDED)

```
 SCHOLAR)

++++++++++++++++++++++++++++++++++++++++++++++++++++++++


 HELLO, MY NAME IS SCHOLAR. I WILL HELP YOU IN REVIEWING
THE GEOGRAPHY OF SOUTH AMERICA. THIS IS ABC REGIONAL
HIGH SCHOOL. TODAY IS 04/29/70.



 IF YOU HAVE CAREFULLY STUDIED YOUR INSTRUCTIONS, WE
MAY BEGIN THE REVIEW NOW.


 WHEN YOU ARE READY TO START, TYPE READY --- REMEMBER
TO TERMINATE WITH AN ASTERISK * FOLLOWED BY A CARRIAGE
RETURN.


*READY*

 PLEASE TYPE YOUR NAME --- MR., MISS, OR MRS. FOLLOWED
BY FIRST NAME, MIDDLE NAME OR MIDDLE INITIAL, AND LAST
NAME.

*MR. JAIME R. CARBONELL*

 NAME OF YOUR INSTRUCTOR, PLEASE:

*MR. BROWN*



 LET'S BEGIN OUR DISCUSSION NOW, MR. CARBONELL.

++++++++++++++++++++++++++++++++++++++++++++++++++++++++


   APPROX WHAT IS THE POPULATION IN GUYANA?

*1500000*

 YOUR ANSWER IS INCORRECT.
 THE CORRECT ANSWER IS: 700000.

 GUYANA IS ----.

*A COUNTRY*

 SELECT AN ALTERNATIVE FROM THE LIST:

 ALTITUDE
 MILES
 COUNTRY
 DEGREES FAHRENHEIT
```

Fig. C.3 (a) On-Line Mixed-Initiative Protocol (CONT)

```
TO COMPLETE THE SENTENCE:

ECUADOR IS A ---.

*COUNTRY*

THAT'S RIGHT.

THE AREA OF ECUADOR IS APPROX --- SQUARE MILES.

*100000*

THAT'S ABOUT RIGHT. MORE ACCURATELY, THE ANSWER IS
     104000


USE ONE OF THE FOLLOWING:

1190400.
-5777216.
422784.
4000000.

TO ANSWER THE QUESTION:

  APPROX WHAT IS THE POPULATION IN ECUADOR?

*1190400.*

YOUR ANSWER IS INCORRECT.

SELECT AN ALTERNATIVE FROM THE LIST:

8000000.
2675500.
3883400.
3418800.

TO FILL THE BLANK BELOW:

THE POPULATION IN VENEZUELA IS APPROX --- PEOPLE.

*8000000.*

VERY GOOD.

  WHAT ARE THE LOCATIONS IN ECUADOR?

*HOW MANY PEOPLE ARE THERE IN ECUADOR?*

  THE POPULATION IS APPROX 4000000. PEOPLE.

NOW ANSWER THE QUESTION YOU DIDN'T ANSWER BEFORE.

*SOUTH AMRICA*
```

Fig. C.3 (b) On-Line Mixed-Initiative Protocol (CONCLUDED)

REFERENCES

1. Carbonell, J.R., Mixed-Initiative Man-Computer Instructional Dialogues, Ph.D. Thesis, Massachusetts Institute of Technology, Department of Engineering, May 31, 1970.

2. Quillian, M.R., Semantic Memory, Ph.D. Thesis, Carnegie Institute of Technology, Pittsburgh, 1966.

3. Quillian, M.R., "The Teachable Language Comprehender: A Simulation Program and Theory of Language," Comm. of the ACM, 12, No. 8, August 1969, pp. 459-476.

4. Cullen, John W., Cowdery, Richard S., Butler, Alfred K. and Harris, Lawrence, "The Development of a Computer-Directed Training Subsystem and Computer Operator Training Material for the Air Force Phase II Base Level System," Report No. ESD-TR-70-27, United States Air Force (Hanscom Field), November 30, 1969.

5. Frye, C.H., "CAI Languages: Capabilities and Applications,' Datamation, 14, No. 9, 1968, pp. 34-37.

6. Zinn, K.L., "Languages for Programming Conversational Use of Computers in Instruction," Proc. Intern'l Fed. Info. Proc., Edinburgh, 1968.

7. Bryan, G.L., "Computers and Education," Computers and Automation, 18, No. 3, 1969, pp. 1-4.

8. Feurzeig, W. and Papert, S. "Programming-Languages As a Conceptual Framework for Teaching Mathematics, Proc. NATO Sc. Conf. on Comp. and Learning, Nice, France, 1968.

9. Swets, J.A., and Fuerzeig, W., "Computer-Aided Instruction," Science, 150, 1965, pp. 572-576.

10. Feingold, S.L., "PLANIT-A Flexible Language Designed for Computer-Human Interaction," Proc. Fall Joint Comp. Conf., 31, 1967, pp. 545-552.

11. Taylor, E.F., "The ELIZA Program: Conversational Tutorial," IEE Annual Convention, New York, March 1967.

12. Chomsky, N., Aspects of the Theory of Syntax, The M.I.T. Press, Cambridge, Mass., 1965.

13. Weizenbaum, J., "ELIZA - A Computer Program for the Study of Natural Language Communications Between Man and Machine," Comm. of the ACM, 9, No. 1, January 1966, pp. 36-45.

14. Rovner, P.D. and Fledman, J.A., "The LEAP Language and Data Structure," Lincoln Lab. Rep. M.I.T., January 1968.

15. Johnson, T.E., "A Mass Storage Relational Data Structure for Computer Graphics and Other Arbitrary Sata Storages," Dep. of Arch. Rept., M.I.T., October 1967.

16. Bobrow, D.G., Murphy, D.L., and Teitelman, W., The BBN-LISP System, Bolt Beranek and Newman Inc., Cambridge, Mass, 1968.

17. Bobrow, D.G., and Murphy, D.L., "Structure of a LISP System Using Two-Level Storage," Comm. ACM, 10, No. 3, March 1967, pp. 155-159.

18. Veliz, C. (ed.), Latin America and the Caribbean: A Handbook, Praeger, New York, 1968.

19. Aguilar, Atlas Medio Universal y de España, Aguilar, Madrid, Spain, 1962.

20. Finch, V.X., Trewartha, G.T., Robinson, A.H., and Hammond, E.H., Elements of Geography, McGraw-Hill, New York, 1957.

21. Miller, G.A., "The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information," Psychological Review, 63, pp. 81-96, 1956.

22. Collins, A.M., and Quillian, M.R., "Retrieval Time from Semantic Memory," J. of Verbal Learning and Verbal Behavior, 8, 1969, pp. 240-247.

23. Minsky, M. (ed), Semantic Information Processing, The M.I.T. Press, Cambridge, Mass., 1968.

24. Carbonell, J.R., "Interactive Non-Deterministic Computer-Assisted Instruction," Proc. Intern'l Symp. on Man-Machine Syst., Cambridge, England, Sept. 1969. See also French version: "L'Enseignment Interactif et Non Directif Assiste Par Ordinateur," Informatique et Gestion, Paris, May 1970, pp. 21-30.

25. Ashby, W.R., Letter to Ed., Science, 168, No. 3933, 15 May 1970, p. 777.

26. Schwarcz, R.M., Burger, J.F., and Simmons, R.F., "A Deductive Question-Answerer for Natural Language Inference," Comm. ACM, 13, No. 3, March 1970, pp. 167-183.

27. Simmons, R.F., personal communication, May 1970.

28. Feigenbaum, E.A., and Feldman, J. (eds.), Computers and Thought, McGraw-Hill, New York, 1963.

29. Bobrow, D.G., "Natural Language Input for a Computer Problem-Solving System," Project MAC Rep. TR-1, M.I.T., Cambridge, 1964.

30. Simmons, R.F., "Natural Language Question-Answering Systems: 1969," Comm. ACM, 13, No. 1, Jan. 1970, pp. 15-30.

31. Minsky, M., "Form and Computer Science," Journal ACM, 17, No. 2, April 1970, pp. 197-215.

32. Green, B.F., Wolf, A.K., Chomsky, C., and Laughery, K., "Baseball. an Automatic Question Answerer," in Computers and Thought, ed. by E.A. Feigenbaum and J. Feldman, McGraw-Hill, New York, 1963.

33. Raphael B., "SIR: A Computer Program for Semantic Information Retrieval," Project MAC Rep. TR-2, M.I.T., Cambridge, Jan. 1970.

34. Kellogg, C.H., "A Natural Language Compiler for On-Line Data Management," Syst. Dev. Corp. SP-3160, Santa Monica, Calif. Aug. 1968.

BIBLIOGRAPHY

Atkinson, R.C., "CAI-Learning Aspects" in Computers and Education, ed. by R.W. Gerard, McGraw-Hill, New York, 1967.

Carbonell, J.R., "On Man-Computer Interaction: A Model and Some Related Issues," IEEE Trans. of Syst. Sc. and Cybernetics, 5, Jan. 1969, pp. 16-26.

Carbonell, J.R., Elkind, J.I., and Nickerson, R.S., "On the Psychological Importance of Time in a Time-Sharing System," Human Factors, 10, 10, No. 2, 1968, pp. 135-142.

Carbonell, J.R., and Klatt, M.M., "LIBRO: An On-Line Examination System for Foreign-Language Instruction," unpublished paper, 1968.

Chomsky, N., Syntactic Structures, Mouton & Co., The Hague, 1957.

Fenichel, R.R., Weizenbaum, J., and Yochelson, J.C., "A Program to Teach Programming," Comm. ACM, 13, No. 3, March 1970, pp. 141-146.

Rockart, J.F., Scot Morton, M.S., and Zannetos, Z.S., "Associative Learning Project - Phase-1 System," Working Paper, A.P. Sloan School of Management, M.I.T., Cambridge, January 1970.

Sheridan, T.B., and Ferrell, W.R., "Human Control of Remote Computer Manipulators," Proc. Intern'l Joint Conf. on Artificial Intelligence, Washington, D.C., 1969.

Simmons, R.F., and Silberman, H.F., "A Plan for Research Toeard Computer-Aided Instruction with Natural English," System Dev. Corp. TM-3623, Santa Monica, Calif., 1967.

Uhr, L., "The Automatic Generation of Teaching Machine Programs," Unpublished paper, 1965.

Uttal, W.R., Pasich, T., Rogers, M., and Hieronymus, R., "Generative Computer-Assited Instruction," Mental Health Res. Ins., Comm. 243, Ann Arbor, 1969.

Wexler, J.D., "A Generative Teaching System That Uses Information Nets and Skeleton Patterns," Unpublished Ph.D. Thesis, Univ. of Wisconsin, Madison, 1970.

Woods, W.A., "Augmented Transition Networks for Natural Language Analysis," Rep. CS-1 to NSF, the Aiken Computation Lab., Harvard University, Cambridge, Mass., December 1969.

Unclassified

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) Bolt Beranek and Newman Inc. 50 Moulton Street Cambridge, Massachusetts 02138	2a. REPORT SECURITY CLASSIFICATION Unclassified 2b. GROUP N/A

3. REPORT TITLE

MIXED-INITIATIVE SYSTEMS FOR TRAINING AND DECISION-AID APPLICATIONS

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)
Final Report (1 July 1969 to 31 August 1970)

5. AUTHOR(S) (First name, middle initial, last name)
Jaime R. Carbonell
Allan M. Collins

6. REPORT DATE November 1970	7a. TOTAL NO. OF PAGES 126	7b. NO. OF REFS 47
8a. CONTRACT OR GRANT NO. F19628-69-C-0298 b. PROJECT NO. c. d.	9a. ORIGINATOR'S REPORT NUMBER(S) ESD-TR-70-373 9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report) BBN Report No. 2054	

10. DISTRIBUTION STATEMENT
1-This document has been approved for public release and sale; its distribution is unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY Deputy for Command & Management Sys. Hq Electronic Systems Div. (AFSC) L.G. Hanscom Field, Bedford, Mass.

13. ABSTRACT

The project reported in this document presents SCHOLAR, the first prototype system capable of a true mixed-initiative man-computer dialogue on a given topic. The computer is capable of answering questions from the man using a comfortable subset of English), as well as of generating questions, analyzing the man's responses, and producing consequent actions. All this occurs without anticipation of conversational items and sequences. A very powerful program, applicable to many subject matters, acts upon a highly structured data base to generate the computer answers and questions, to evaluate the man's answers, and to produce suitable action sequences.

The U.S. Air Force can benefit considerably from the development of such mixed-initiative "knowledgeable" systems. The relevancy for applications like training, logistics and resource allocation, command and control systems, intelligence systems, and on-line design and planning is clear. A system built along the lines of SCHOLAR can be valuable as an on-line aid to decision makers, by facilitating the interaction with complex and highly structured military data bases. SCHOLAR is also ideally suited to evolve into an on-line training facility to assist computer users in utilizing a new computer systm or language.

DD FORM 1 NOV 65 1473 (PAGE 1)

S/N 0101-807-6811

Unclassified

Security Classification

A-31408

Unclassified

Security Classification

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Computer-assisted instruction						
Artificial intelligence						
Man-computer interaction						
Natural-language comprehension						
Natural-language generation						
Semantic networks						
Mixed-initiative dialogues						
Decision Aids						
Performance Aids						

DD FORM 1 NOV 65 1473 (BACK)

S/N 0101-807-6821

Unclassified

Security Classification

A-31409